THEORY AND APPLICATION OF THE MOMENTARY FOURIER TRANSFORM

by

Sandor Albrecht

M.Sc.E.E. Technical University of Budapest, Hungary, 1993

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

September 1998

Department of Electrical and Computer Engineering

The University of British Columbia
Vancouver, Canada

Date October 5, 1998

Abstract

The discrete Fourier transform (DFT) is a widely used tool in signal or image processing and its efficiency is important. There are applications where it is desirable to use relatively small, successive, overlapped DFTs to obtain the spectrum coefficients. The momentary Fourier transform (MFT) computes the DFT of a discrete-time sequence for every new sample in an efficient recursive form. In this thesis we give an alternate derivation of the MFT using the momentary matrix transform (MMT). Recursive and non-recursive forms of the inverse MFT are also given, which can provide efficient frequency domain manipulation (*e.g.* filtering). Discussion on the properties and examples of the usage of the MFT is given, followed by a survey on its efficiency.

In this work we investigate the applicability of the MFT to synthetic aperture radar (SAR) signal processing, and in particular show what advantages the MFT algorithm offers to the SPECtral ANalysis (SPECAN) method and burst-mode data processing. In the SPECAN algorithm, the received signals are multiplied in the time domain by a reference function, and overlapped short length DFTs are used to compress the data. The azimuth FM rate of the signal varies in each range cell, which leads to the issue of keeping the azimuth resolution and output sampling rate constant. After the introduction to SPECAN, we show what advantages and disadvantages the MFT has compared to the FFT algorithms.

When a SAR system is operated in burst-mode, its azimuth received signal has a segmented frequency-time energy in its Doppler history. It requires that IDFTs be located at specific points in the spectral domain to perform the azimuth signal compression. After the introduction of the burst-mode data properties, we show why the short IFFT (SIFFT) algorithm has the requirement of arbitrary-length, highly-overlapped IDFTs to process burst-mode data, in which case the IMFT is shown to have computational advantages.

Contents

List of Tables

List of Figures

Acknowledgements

First of all, I would like to thank my Mother and Edina for providing consistent support and encouragement throughout my studies at UBC. Without their love and care this work would not have been completed.

I would like to thank my supervisor, Dr. Ian Cumming, for his supervision and academic guidance, as well as providing the opportunity to continue my research of momentary Fourier transform in the field of synthetic aperture radar processing.

I am grateful for the financial support provided for this research by the Natural Sciences and Engineering Council of Canada and by MacDonald Dettwiler and Associates.

Finally, I would like to thank all of the members of the Radar Remote Sensing Group at UBC for providing a friendly and effective work environment.

Chapter 1

Introduction

1.1 Background

Linear transformations, such as the discrete Fourier transform (DFT) are frequently used in digital signal processing, and their efficiency is very important. In applications where the DFT is applied to a signal, it is often desirable to use successive, possibly overlapping DFTs of smaller extent than the full length of the signal to obtain the spectrum coefficients. These transformations are normally off-line operations on blocks of data, requiring *N* samples of the signal before the transformation can be computed. The momentary Fourier transform (MFT) which is derived here is a method of computing the DFT of a sequence in incremental steps. It can be computed using an efficient recursive formula, and it is useful in cases where the detailed evolution of the spectra of a discrete series is wanted, and in cases where only a few Fourier coefficients are needed.

The spectrum components of the MFT can be calculated independently and only one complex multiplication and two complex additions are needed to update each spectrum component. The inverse momentary Fourier transform (IMFT) is the dual of the MFT and shares the same property, while the non-recursive form of the IMFT requires only additions to obtain a sample of the time sequence from its spectrum with N samples delay.

The computational order of the MFT to update an N-point DFT is N, a factor of log_2N improvement over the radix-2 FFT algorithm if all incremental results are needed. If only a sub-set of the transform domain components are needed, the computing load of the MFT can be further reduced, calculating only the coefficients of interest. The MFT does not rely upon on N being power of two to obtain its efficiency, in contrast to standard FFT algorithms.

Uses of the incremental DFT were introduced by Papoulis in 1977 [1], and by Bitmead and Anderson in 1981 [5]. A detailed derivation of the momentary Fourier transform was given by Dudás in 1986 [6]. In 1991, Lilly gives a similar derivation, using the term "moving Fourier transform", and uses the MFT for updating the model of a time-varying system [7]. In this thesis we further develop the theory of MFT, examine its applications and in particular, see what advantages it offers to synthetic aperture radar data processing.

A synthetic aperture radar (SAR) is a powerful sensor in remote sensing which is capable of observing geophysical parameters of the Earth's surface, regardless of time of day and weather conditions [3]. SAR systems are extensively used for monitoring ocean surface patterns, sea-ice cover, agricultural features and for military applications such as in the detection and tracking of moving targets. A SAR transmits radar signals from an airborne or spaceborne antenna which is perpendicular to the flight direction of the platform which is travels at a constant velocity. The back-scattered signal is collected by the antenna and stored in a raw format. Extensive signal processing is required to produce the output SAR image.

The SPECtral ANalysis (SPECAN) SAR processing algorithm was developed in 1979 by MacDonald Dettwiler and Associates, as a multi-look version of the deramp-FFT method of pulse compression. In SPECAN, the received signals are multiplied in the time domain by a reference function, and overlapped short length DFTs are used to compress the data. In contrast, a precision processing algorithm such as the Range Doppler (RD) method requires both forward and inverse DFT operations, thus it is less computationally efficient. SPECAN is an efficient algorithm for moderate to low resolution processing and generally implemented in quick look processors for viewing of magnitude detected imagery data.

Burst-mode operation is used in SAR systems, to image wide swaths, to save power or save data link bandwidth. Several spaceborne remote sensing missions employ the ScanSAR mode in addition to other operational modes for radar imaging. Canada's Radarsat satellite, which was successfully launched in 1995, is a sophisticated Earth observation system developed to monitor environmental changes. The imaging platform supports various SAR operating modes, including a ScanSAR mode for the low-resolution (~100m) imaging of ground regions of width 500 km.

An Advanced SAR (ASAR) system will be flown on the Envisat satellite polar platform to be launched in 2000 by the European Space Agency. This system will be able to operate in three burst-modes: alternating polarization mode (AP), wide swath mode (WS) and global monitoring mode (GM). Alternating polarization mode provides high resolution in any swath with polarization changing from sub-aperture to sub-aperture within the synthetic aperture. This results in two images of the same scene in different polarizations combination with approximately 30 m resolution. In the wide swath mode the ScanSAR technique is used providing images of a wider swath (405 km) with medium resolution (150 m).

1.2 Thesis objectives and outline

The objective of this research is to further develop the theory of the MFT, examine its properties and applications, and in particular, see what advantages it offers to SPECAN processing and to the short IFFT (SIFFT) burst-mode processing algorithm.

Chapter 2 presents the theory and properties of the momentary Fourier transform. Here, we introduce the recursive form of the momentary matrix transform (MMT), and show when the MMT takes the form of the DFT or the IDFT, the resulting MFT and IMFT have an efficient computational structure. The properties and computing efficiency of the MFT is also discussed in this chapter.

In Chapter 3, an overview of SAR processing is given, where the conventional compression method of the SAR signals is studied. This chapter gives a background knowledge for the research of the SPECAN algorithm and burst-mode data processing.

The azimuth FM rate of the received signal varies in each range cell, which leads to the issue of keeping the azimuth resolution and output sampling rate constant. After the introduction of the SPECAN algorithm in Chapter 4, we show what advantages the MFT method offers vs. the FFT algorithms when they are applied to the SPECAN SAR processing algorithm.

In Chapter 5, the ScanSAR operation mode will be introduced, and the received burst-mode data properties will be analyzed. After the effect of the varying SAR parameters and SNR/efficiency tradeoffs, a survey on the arithmetic of the SIFFT algorithm using IMFT and IFFT is given. Here, we show that the IMFT algorithm can improve the computational efficiency of the SIFFT algorithm in certain burst-mode data processing cases.

Finally in Chapter 6, conclusions of the efficiency and applicability of the momentary Fourier transform to SAR processing will be drawn, and suggestions for possible future work will be given.

Chapter 2

Theory and Properties of the Momentary Fourier Transformation

2.1 Introduction

In this Chapter, we give a derivation of the momentary Fourier transforms from the momentary matrix transform in Section 1.2. Section 1.3 gives a survey on the properties of the MFT, and in Section 1.4 a discussion on its computational efficiency is given.

2.2 The Momentary Fourier Transformations derived form the Recursive Momentary Matrix Transformation

In this section, we introduce the matrix form of the momentary transform, and show that it has a recursive form. We also show that when the momentary matrix transform takes the form of the DFT or the inverse DFT, the resulting MFT has an efficient (recursive) computational structure. In the last part of the section, the inverse of the MFT is introduced, as well.

2.2.1 The Recursive Momentary Matrix Transformation

Let x_i be a sample of an arbitrary complex-valued sequence of one variable. The sequence will be analyzed through an N-point window, ending at the current sample i. In subsequent analyses, the window will be advanced one sample at a time. At sample i x_i enters the window, while x_{i-N} leaves the window, as shown in Figure 1.

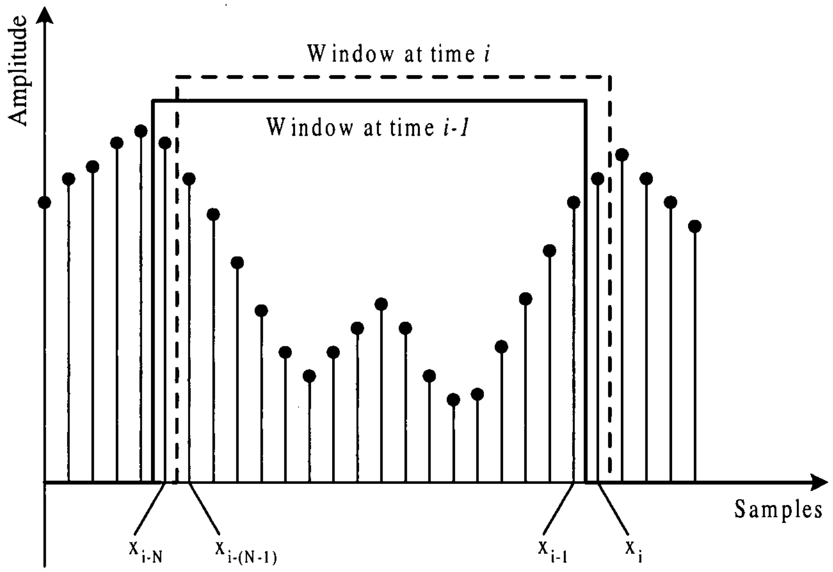


Figure 1 Windowing of the discrete-time function

At samples $i\text{-}1$ and i, the windowed function can be represented by the following two column vectors:

$$\mathbf{x}_{i\text{-}1} = \begin{bmatrix} x_{i\text{-}N} \\ . \\ . \\ . \\ x_{i-1} \end{bmatrix}, \qquad \mathbf{x}_i = \begin{bmatrix} x_{i-(N-1)} \\ . \\ . \\ x_{i-1} \\ x_i \end{bmatrix}$$

(1)

Let **T** be an *NxN* non-singular matrix, which represent a linear transformation and has the inverse $\mathbf{T}^{-1}$. The sequence of index vectors is transformed by **T** at each sample:

$$\ldots,\ \mathbf{y}_{i\text{-}1} = \mathbf{T}\mathbf{x}_{i-1},\ \ \mathbf{y}_i = \mathbf{T}\mathbf{x}_i,\ \ \ldots \tag{2}$$

Let **P** be the *NxN* elementary cyclic permutation matrix. When the vector $\mathbf{x}_{i\text{-}1}$ is multiplied by **P**, a one-element circular shift is performed, such that the index of each element is increased by one, and the first element becomes the last one:

$$\mathbf{P}\mathbf{x}_{i-1} = \begin{bmatrix} x_{i\text{-}(N\text{-}1)} \\ . \\ . \\ x_{i-1} \\ x_{i-N} \end{bmatrix},\ \text{where}\ \ \mathbf{P} = \begin{bmatrix} 0 & 1 & . & . & 0 \\ . & 0 & 1 & . & 0 \\ . & . & 0 & 1 & . \\ . & . & . & 0 & 1 \\ 1 & 0 & . & . & 0 \end{bmatrix} \tag{3}$$

Using the result above, the $\mathbf{x}_i$ vector can be expressed by the shifted $\mathbf{x}_{i\text{-}1}$ vector and with an adjustment vector $\Delta\mathbf{x}_i$ made from the difference between the samples entering and leaving the window:

$$\mathbf{x}_i = \begin{bmatrix} x_{i\text{-}(N\text{-}1)} \\ . \\ . \\ x_{i-1} \\ x_{i-N} \end{bmatrix} + \begin{bmatrix} 0 \\ . \\ . \\ 0 \\ x_i\text{-}x_{i-N} \end{bmatrix} = \mathbf{P}\,\mathbf{x}_{i-1} + \Delta\mathbf{x}_\mathrm{i} \tag{4}$$

Substituting (4) into the transformation associated with the *ith* window in (2) and using the inverse transform $\mathbf{x}_{i-1} = \mathbf{T}^{-1}\, \mathbf{y}_{i-1}$, the following relationships are obtained:

$$\mathbf{y}_i = \mathbf{T}\,\mathbf{x}_i = \mathbf{T}\left[\mathbf{P}\,\mathbf{x}_{i-1} + \Delta\mathbf{x}_i\right] = \mathbf{T}\,\mathbf{P}\,\mathbf{T}^{-1}\mathbf{y}_{i-1} + \mathbf{T}\,\Delta\mathbf{x}_{\mathrm{i}} \tag{5}$$

Equation (5) expresses the *recursivity of the momentary matrix transforms (MMT)*, since calculation of the newly transformed index vector $\mathbf{y}_{\mathrm{i}}$ needs the previously transformed index vector $\mathbf{y}_{\mathrm{i-1}}$ and the difference between samples entering and leaving the window.

2.2:2 The diagonal form of the MMT

The momentary matrix transform is particularly efficient and it can be calculated by components only if the product of similarity matrix transform $\mathbf{TPT}^{-1}$ in (5) is diagonal. The $\mathbf{P}$ matrix has N distinct eigenvalues ($\lambda_0, \ldots, \lambda_{N-1}$) which are the *nth* complex unit roots, $\lambda_k = w^{-k} = e^{j2\pi\, k/N}, k = 0,1,2,\ldots N-1$. To each eigenvalue, N linearly independent eigenvector corresponds as follows:

$$\lambda_0 = w^0 = 1 \;\Leftrightarrow\; \mathbf{s}_0 = \begin{bmatrix} 1 \\ 1 \\ . \\ . \\ 1 \end{bmatrix};\quad \lambda_1 = w^{-1} \;\Leftrightarrow\; \mathbf{s}_1 = \begin{bmatrix} 1 \\ w^{-1} \\ w^{-2} \\ . \\ w^{-(N-1)} \end{bmatrix};\ \ldots$$

$$\lambda_k = w^{-k} \Leftrightarrow \mathbf{s}_k = \begin{bmatrix} 1 \\ w^{-k} \\ w^{-2k} \\ . \\ w^{-(N-1)k} \end{bmatrix} ; \; ... \; ; \lambda_{N-1} = w^{-N-1} \Leftrightarrow \mathbf{s}_{N-1} = \begin{bmatrix} 1 \\ w^{-(N-1)} \\ w^{-2(N-1)} \\ . \\ w^{-(N-1)(N-1)} \end{bmatrix} \tag{6}$$

If the eigenvectors are chosen to be the columns of the inverse of the **T** matrix, then $\mathbf{TPT}^{-1}$ is a diagonal matrix, with the eigenvalues of **P** along its diagonal:

$$\mathbf{TPT}^{-1} = \mathbf{S}^{-1}\mathbf{PS} = \begin{bmatrix} | & | & & | & | \\ | & | & & | & | \\ \mathbf{s}_0^{-1} & \mathbf{s}_1^{-1} & ... & \mathbf{s}_{N-2}^{-1} & \mathbf{s}_{N-1}^{-1} \\ | & | & & | & | \\ | & | & & | & | \end{bmatrix} \mathbf{P} \begin{bmatrix} | & | & & | & | \\ | & | & & | & | \\ \mathbf{s}_0 & \mathbf{s}_1 & ... & \mathbf{s}_{N-2} & \mathbf{s}_{N-1} \\ | & | & & | & | \\ | & | & & | & | \end{bmatrix} = \begin{bmatrix} \lambda_0 & 0 & . & . & 0 \\ 0 & \lambda_1 & 0 & . & 0 \\ . & . & \lambda_2 & 0 & . \\ . & . & . & . & 0 \\ 0 & 0 & . & . & \lambda_{N-1} \end{bmatrix} \tag{7}$$

where **S** is the eigenvector matrix of **P**.

The diagonalizing matrix **S** is not unique. An eigenvector $\mathbf{s}_k$ can be multiplied by a constant, and will remain an eigenvector [2]. Therefore the columns of **S** can be multiplied by any nonzero constants and produce a new diagonalizing **S**. There is also no preferred order of the columns of **S**. The order of the eigenvectors in **S** and the eigenvalues in the diagonal matrix is automatically the same. Therefore, all **T** matrixes, which satisfy the above mentioned properties will diagonalize the momentary matrix transform:

$$\mathbf{y}_i = \begin{bmatrix} \lambda_k & 0 & . & . & 0 \\ 0 & \lambda_l & 0 & . & 0 \\ . & . & . & 0 & . \\ . & . & . & . & 0 \\ 0 & 0 & . & . & \lambda_m \end{bmatrix} \mathbf{y}_{i-1} + \mathbf{T}_{N\text{-}1}\left(x_i - x_{i\text{-}N}\right),$$

(8)

where *k, l, m* ∈ *{0,1,…,N-1}* and $\mathbf{T}_{N\text{-}1}$ is the last column of the **T** matrix.

2.2.3 Inverse of the diagonalized MMT

If $\mathbf{y}_i$ is available at each sample and the columns of **T** are the eigenvectors of **P**, an efficient implementation of the inverse of the MMT can be obtained. The *inverse MMT (IMMT)* at time *i*:

$$\mathbf{x}_i = \mathbf{T}^{-1}\mathbf{y}_i$$

(9)

$$\begin{bmatrix} x_{i\text{-}(N\text{-}1)} \\ . \\ . \\ . \\ x_{i\text{-}1} \\ x_i \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \end{bmatrix} \begin{bmatrix} y_{i,0} \\ y_{i,1} \\ . \\ . \\ . \\ y_{i,N\text{-}1} \end{bmatrix}$$

(10)

The first row of $\mathbf{T}^{-1}$ contains only ones (10), so the oldest element of $\mathbf{x}_i$ can be computed using adds only:

$$x_{i-(N-1)} = \sum_{k=0}^{N-1} y_{i,k}$$

(11)

from which the elements of the input sequence $(x_{i-(N-1)}\ldots x_i)$ can be computed from the transform domain sequence $\mathbf{y}_i$ with $N\text{-}1$ sample delay.

2.2.4 Momentary Fourier Transform

The matrix of the Discrete Fourier Transform (DFT) and the Inverse Discrete Fourier Transform (IDFT) have the properties described in Section 1.1.2, thus their columns are the eigenvectors of the matrix **P**:

$$\mathbf{DFT} = \mathbf{F} = \mathbf{S}^{-1} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & w & w^{2} & . & . & w^{N-1} \\ 1 & w^{2} & w^{4} & . & . & w^{2\cdot(N-1)} \\ 1 & . & . & . & . & . \\ 1 & . & . & . & . & . \\ 1 & w^{N-1} & w^{2\cdot(N-1)} & . & . & w^{(N-1)(N-1)} \end{bmatrix} \tag{12}$$

$$\mathbf{IDFT} = \mathbf{F}^{-1} = \mathbf{S} = \frac{1}{N}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & w^{-1} & w^{-2} & . & . & w^{-(N-1)} \\ 1 & w^{-2} & w^{-4} & . & . & w^{-2\cdot(N-1)} \\ 1 & . & . & . & . & . \\ 1 & . & . & . & . & . \\ 1 & w^{-(N-1)} & w^{-2\cdot(N-1)} & . & . & w^{-(N-1)(N-1)} \end{bmatrix} =$$

$$\frac{1}{N}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & w^{N-1} & . & . & w^{2} & w \\ 1 & w^{2(N-1)} & . & . & w^{4} & w^{2} \\ 1 & . & . & . & . & . \\ 1 & . & . & . & . & . \\ 1 & w^{(N-1)(N-1)} & . & . & w^{2(N-1)} & w^{N-1} \end{bmatrix} \tag{13}$$

Using that w is the *Nth* complex unit root (i.e. $w^{-k} = w^{N-k}$), it can be seen that the columns of the IDFT matrix are the same as the DFT matrix, but they are in reverse order from the second column (13). Therefore, if $\mathbf{T}$ performs the DFT (14) or the IDFT (15), diagonal forms of the MMT can be obtained:

$$\mathbf{y}_i = \mathbf{F}\,\mathbf{P}\,\mathbf{F}^{-1}\,\mathbf{y}_{i\text{-}1} + \mathbf{F}\,\Delta\mathbf{x}_i = \begin{bmatrix} 1 & 0 & . & . & 0 \\ 0 & w^{-1} & 0 & . & 0 \\ . & . & w^{-2} & 0 & . \\ . & . & . & . & 0 \\ 0 & 0 & . & . & w^{-(N-1)} \end{bmatrix} \mathbf{y}_{i-1} + \begin{bmatrix} 1 \\ w^{-1} \\ w^{-2} \\ . \\ w^{-(N-1)} \end{bmatrix} (x_i - x_{i\text{-}N}) \tag{14}$$

$$\mathbf{x}_i = \mathbf{F}^{-1}\,\mathbf{P}\,\mathbf{F}\,\mathbf{x}_{i\text{-}1} + \mathbf{F}^{-1}\,\Delta\mathbf{y}_i = \begin{bmatrix} 1 & 0 & . & . & 0 \\ 0 & w^{1} & 0 & . & 0 \\ . & . & w^{2} & 0 & . \\ . & . & . & . & 0 \\ 0 & 0 & . & . & w^{N-1} \end{bmatrix} \mathbf{x}_{i-1} + \begin{bmatrix} 1 \\ w^{1} \\ w^{2} \\ . \\ w^{N-1} \end{bmatrix} (y_i - y_{i\text{-}N}) =$$

$$= \begin{bmatrix} 1 & 0 & . & . & 0 \\ 0 & w^{-(N\text{-}1)} & 0 & . & 0 \\ . & . & w^{-(N\text{-}2)} & 0 & . \\ . & . & . & . & 0 \\ 0 & 0 & . & . & w^{-1} \end{bmatrix} \mathbf{x}_{i\text{-}1} + \begin{bmatrix} 1 \\ w^{-(N-1)} \\ w^{-(N\text{-}2)} \\ . \\ w^{-1} \end{bmatrix} (y_i - y_{i-N}) \tag{15}$$

Equation (14) expresses the recursive equation of the *momentary Fourier-transform (MFT)*. The N-element vector $\mathbf{y}_\mathrm{i}$ contains the Fourier coefficients of the N-point sequence $\mathbf{x}_\mathrm{i}$ ending at sample i. Note that each spectral component $y_{i,k}$ can be calculated independently,

$$y_{i,k} = w^{-k}\left(y_{i-1,k} + x_i - x_{i-N}\right)$$

(16)

which increases efficiency if only a few frequency components need to be computed, as in the zoom transform.

On the other hand, equation (15) is the dual pair of the MFT, the recursive *inverse momentary Fourier-transform (IMFT)*, where the N-element vector $\mathbf{x}_i$ contains the N-point time sequence and $\mathbf{y}_i$ contains N Fourier coefficients ending at frequency bin i. Note that the each sample in $\mathbf{x}_i$ can also be obtained independently and the same twiddle factors, but in different order, can be used to calculate both the MFT and IMFT.

Thus it has been showed that if the DFT or the IDFT performs the momentary matrix transform of a sequence the elements of the transformed sequence can be computed recursively and independently using N complex multiplies and $N+1$ complex adds (computational savings are available if the input sequence is real-valued).

2.2.5 The non-recursive Inverse MFT

The non-recursive *inverse momentary Fourier transform* can be expressed using (11) and (13) as follows:

$$x_{i-(N-1)} = \frac{1}{N}\sum_{k=0}^{N-1} y_{i,k}$$

(17)

from which each sample of the input sequence (**x**$_i$) can be computed using adds only from the spectrum (**y**$_i$) with *N-1* sample delay. In this way the MFT- non-recursive IMFT transform pair can provide an efficient frequency-domain manipulation method (e.g. filtering), especially if many of the DFT coefficients are not calculated.

If the elements of **x**$_i$ are real, taking advantage of the conjugate symmetry of the spectrum, the oldest element can be computed using only the real part of the spectrum components:

$$x_{i-(N-1)} = \frac{1}{N}\sum_{k=0}^{N-1} \mathrm{Re}\{y_{i,k}\} \tag{18}$$

It has been showed in [6] that if **x**$_i$ is real, the Hilbert transform of $x_{i-(N-1)}$ can be obtained to sum only·the imaginary part of the spectrum components:

$$H\{x_{i-(N-1)}\} = \frac{1}{N}\sum_{k=0}^{N-1} \mathrm{Im}\{y_{i,k}\} \tag{19}$$

In this case the MFT/non-recursive IMFT pair can be useful for different signal processing applications where the in-phase (I) and quadrature component (Q) of the signal is needed (i.e. communications and radar systems).

2.3 Properties of the MFT

In this section, some properties of the MFT are given. Section 1.3.1 shows how to implement cosine windows in the frequency domain using the MFT. In Section 1.3.2, discussion on the software and hardware implementation of the MFT algorithm is given, while section 1.3.3 gives an example of the use of the MFT.

2.3.1 Cosine windows using MFT

The N-point DFT treats the data sequence as if it has a periodicity of N samples, $x_i = x_{i+kN}$, for all integer k. In practice many signals do not have the above periodicity. If the boxcar window is applied to such a signal, the DFT will treat it if there were discontinuities at its edges. Ringing effects near the edges of filtered signals may occur as a result of these spurious discontinuities [4]. Such effects can be reduced by applying a more appropriate window. In addition to selecting a portion of the input sequence, the window modifies this portion to make it continuous at the edges when regarded as periodically repeated. Several types of window have been described in the literature [1], [4]. This section introduces how the Hanning, Hamming and Blackman window can be implemented using the MFT.

Given the discrete-time sequence $\mathbf{x}_\mathrm{i}$, we wish to calculate the MFT of the windowed data $\boldsymbol{x}_{w,i} = \boldsymbol{w}_i \cdot \boldsymbol{x}_i$ at time i, where $\boldsymbol{w}_i$ is the window function. The *ith* element of the window may be expressed as follows:

Hanning: $$w_i = 0.5\left[1-\cos\left(\frac{2\pi i}{N}\right)\right]$$

Hamming: $w_i = 0.54 - 0.46\cos\left(\frac{2\pi i}{N}\right)$

Blackman: $w_i = 0.42 - 0.5\cos\left(\frac{2\pi i}{N}\right) + 0.08\cos\left(\frac{4\pi i}{N}\right)$

(20)

The derivation for the Blackman window is given below:

$$\begin{aligned}\mathbf{x}_{wi} = \mathbf{w}_i\mathbf{x}_i = &\left[0.42 - 0.5\cos\left(\frac{2\pi i}{N}\right) + 0.08\cos\left(\frac{4\pi i}{N}\right)\right]\mathbf{x}_i = \\ &=0.42\,x_i - 0.25\left[\exp\left(j\frac{2\pi i}{N}\right) + \exp\left(-j\frac{2\pi i}{N}\right)\right]\mathbf{x}_i + \\ &0.04\left[\exp\left(j\frac{4\pi i}{N}\right) + \exp\left(-j\frac{4\pi i}{N}\right)\right]\mathbf{x}_i\end{aligned}$$

(21)

Taking the DFT of each part of (17) the spectrum of the windowed data at time *i*:

$$\mathbf{y}_{wi,\,k} = 0.42\mathbf{y}_{i,\,k} - 0.25\left[\mathbf{y}_{i,\,k+1} + \mathbf{y}_{i,k\text{-}1}\right] + 0.04\left[\mathbf{y}_{i,\,k+2} + \mathbf{y}_{i,k\text{-}2}\right]$$

(22)

Therefore, the MFT of the windowed data can be obtained simply by maintaining the MFT of the non-windowed data and applying a weighted average in the spectrum.

Similar results can be easily derived for the other cosine windows. These are:

Hanning: $y_{wi,k} = 0.5\,y_{i,k} - 0.25\left[y_{i,k+1} + y_{i,k-1}\right]$

Hamming: $y_{wi,k} = 0.54\,y_{i,k} - 0.23\left[y_{i,k+1} + y_{i,k-1}\right]$

(23)

Although only generalized cosine windows can be applied easily with the MFT, arbitrary windows can be approximated if enough cosine terms are used. Note, that the memory requirement of the MFT algorithm gets larger, while its efficiency drops as the number of terms increases.

The edge effect of the boxcar window can also be compensated if the non-weighted moving average of the spectrum components is used. The moving average of a spectrum coefficient at times *i*, for L (L < N) consecutive samples can be expressed as:

$$y_{mavg\_i,k} = \frac{1}{L} \sum_{j=i-L+1}^{i} y_{i,k}$$

(24)

It also has a recursive form where, the calculation of the averaged spectrum coefficient needs the previously calculated average and the difference between the spectrum coefficient entering and leaving the averaging window:

$$y_{mavg\_i+1,k} = y_{mavg\_i,k} + \left(\frac{y_{i+1,k}}{L} - \frac{y_{i-L+1,k}}{L} \right)$$

(25)

The above defined moving average with the long-term average (26) of the MFT coefficients can also be useful for statistical analysis of the input discrete sequence.

$$y_{k,avg} = \frac{1}{M} \sum_{i=1}^{M} y_{i,k}\,, \qquad \mathrm{M >> N}$$

(26)

2.3.2 Implementation of the MFT algorithm

In this section, discussion on the software and hardware implementation of the MFT algorithm is given. Section 1.2.2.1 shows the computer coding of the MFT with its memory requirement, while the principle hardware structure of the MFT is given in Section 1.2.2.2.

2.3.2.1 Software Implementation of the MFT

As it was shown earlier, the spectrum components in the MFT algorithm can be calculated independently from each other. Thus, the MFT can be built up from identical blocks, where a block refers to equation (10). The software implementation of one MFT block can be obtained using the trigonometric form of the equation:

$$y_{i,k} = w^{-k}\left(y_{i-1,k} + x_i - x_{i-N}\right)$$

$$w^{-k} = e^{\frac{-j2\pi k}{N}} = \cos\left(\Phi_k\right) + j\sin\left(\Phi_k\right)$$
$$where \quad {}_k = \frac{-j2\pi k}{N}$$

(27)

$$\mathrm{Re}\{y_{i,k}\} = \cos(\Phi_k)(\mathrm{Re}\{y_{i-1,k}\} + \mathrm{Re}\{x_i - x_{i-N}\}) - \sin(\Phi_k)(\mathrm{Im}\{y_{i-1,k}\} + \mathrm{Im}\{x_i - x_{i-N}\})$$
$$\mathrm{Im}\{y_{i,k}\} = \cos(\Phi_k)(\mathrm{Im}\{y_{i-1,k}\} + \mathrm{Im}\{x_i - x_{i-N}\}) + \sin(\Phi_k)(\mathrm{Re}\{y_{i-1,k}\} + \mathrm{Re}\{x_i - x_{i-N}\})$$

(28)

Equation (28) corresponds to the *kth* MFT block for complex x_i, where $y_{i,k}$ is the *kth* spectrum component at sample *i*. Re{ } means real part and Im{ } means imaginary part of a complex number.

The MFT blocks can be organized in a *for* loop to calculate the needed DFT coefficients. The following pseudo-code segment illustrates the computer coding of the MFT algorithm, assuming the sine and cosine arrays (twiddle factors) have been precomputed:

```
calculate($x_i$-$x_{i-N}$);

for k = start to start+$N_c$-1 do
  MFTblock(k);
endfor
```

If the calculation of the spectrum coefficients is off-line, the difference of the entering and leaving samples of the window can be calculated for the whole data set and stored in a file or an array in the memory. If it is on-line, a modulo-*N* array is needed to calculate x_i-x_{i-N}. The index of the *for* loop in the pseudo-code indicates, that within the valid spectrum components only a smaller interval of the DFT coefficients can be computed. If the parameter *start* is zero and $N_c = N$, then all the spectrum coefficients are going to be calculated. Within the *MFTblock* procedure, the previously computed spectrum coefficient should be stored in an array for the computation of the recent one. The following table gives the memory requirement of the MFT algorithm:

Array type	Size
Twiddle factors (sine and cosine arrays)	$2 \cdot N_c \cdot B$ bits
Modulo-*N* FIFO for complex x_i-x_{i-N}	$2 \cdot N \cdot B$ bits
Spectrum coefficients at time *i-1* - $y_{i-1,k}$	$2 \cdot N_c \cdot B$ bits
Spectrum coefficients at time *i* - $y_{i-1,k}$	$2 \cdot N_c \cdot B$ bits

Table 1 Memory requirement of the MFT

In Table 1, parameter B is the number of bits used during the arithmetic operations, thus $B = 32bits$ if floating point arithmetic is used. B should be at least 24 bits for the fix point arithmetic, concerning the sensitivity of the MFT algorithm for the quantization error of the sine and cosine function.

Note, the memory requirement of the MFT depends on the calculated spectrum coefficients. If the whole spectrum is computed, $N{\cdot}B$ *byte* memory is needed for the computation.

2.3.2.2 Hardware Implementation of the MFT

The trigonometric form of the MFT for one spectrum component (29) can be easily implemented in hardware. From the basic blocks of MFT a parallel hardware structure can be built for the computation of the DFT coefficients. Figure 2 illustrates the block diagram of the concurrent implementation of the MFT blocks of the full MFT algorithm. Note, the updating time of the fully concurrent implementation is equivalent to the propagation time of one MFT block, regardless of the number of the calculated spectrum coefficients.

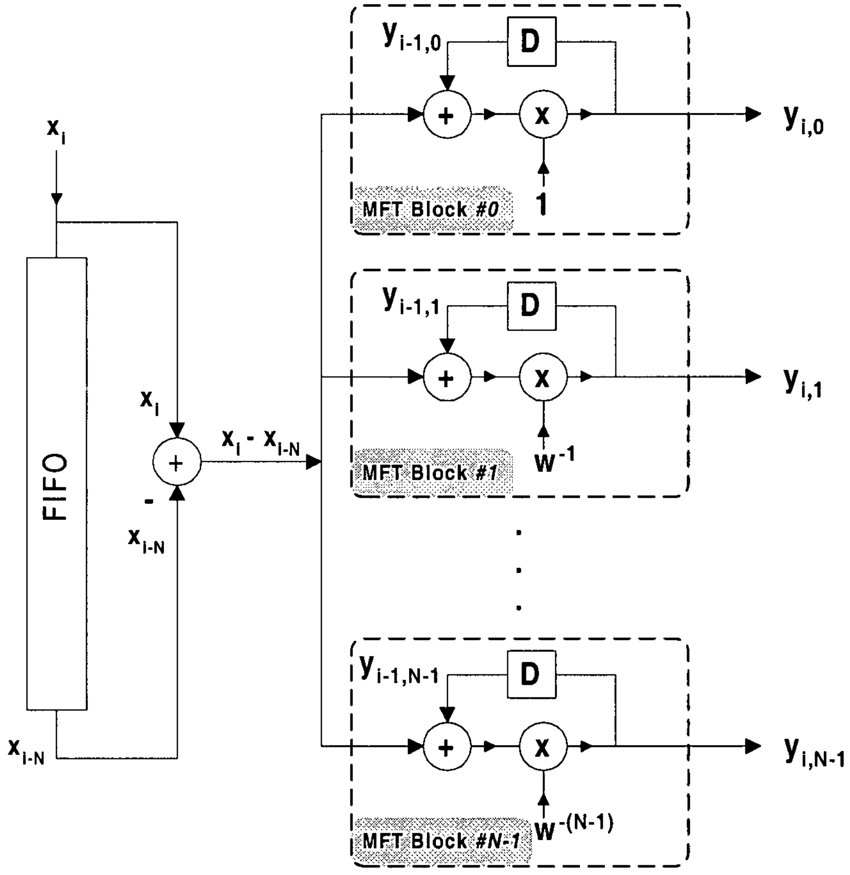


Figure 2 Block diagram of the full MFT algorithm

In Figure 2 the architecture of the full MFT contains a modulo-*N* FIFO register to obtain x_{i-N}. If all the spectrum coefficients are computed, the leaving sample of the window at time *i* can be expressed using the IMFT algorithm:

$$x_{i-N} = \frac{1}{N} \sum_{k=0}^{N-1} y_{i-1,k} \tag{29}$$

Substituting (29) to (10), the recursive equation of one MFT coefficients becomes the following:

$$y_{i,k} = w^{-k}\left(y_{i-1,k} + x_i - \frac{1}{N}\sum_{k=0}^{N-1} y_{i-1,k} \right) \tag{30}$$

In (30), the data sample at time *i* and all the spectrum coefficients at time *i-1* are needed to obtain $y_{i,k}$. The memory requirement of the MFT algorithm is reduced by $2{\cdot}N{\cdot}B$ bits, because there is no need to save the input data samples in a FIFO, while the arithmetic of the MFT increased by *2N-1* real operations due to the calculation of IMFT. The block diagram of the MFT corresponding to equation (30) is shown in Figure 3.

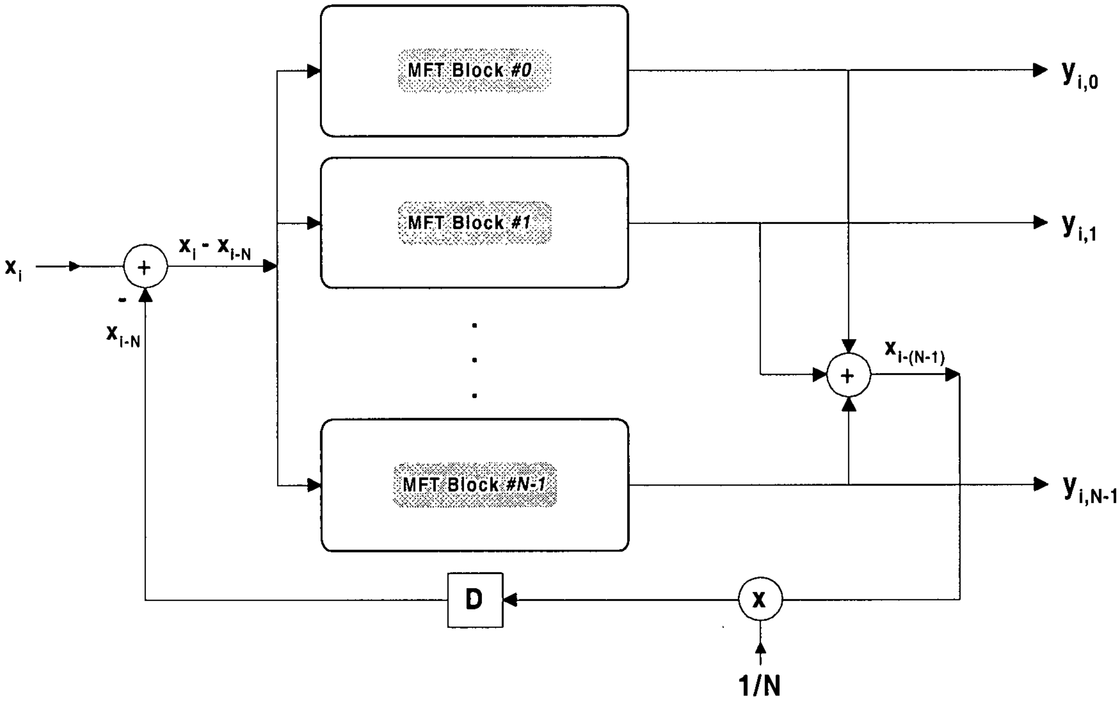


Figure 3 Block diagram of the full MFT algorithm without the modulo-*N* FIFO

The detailed hardware structure of one MFT block for complex x_i is given in Figure 4. This implementation contains four multipliers (MPY) and four adders (ALU) to obtain the complex arithmetic of the MFT. The twiddle factors and the previously calculated spectrum coefficients are stored in registers. Because of the parallel computation of the real and the imaginary part of the MFT coefficients, the updating time of the spectrum coefficients is limited only by the propagation time of two multipliers and two adders.

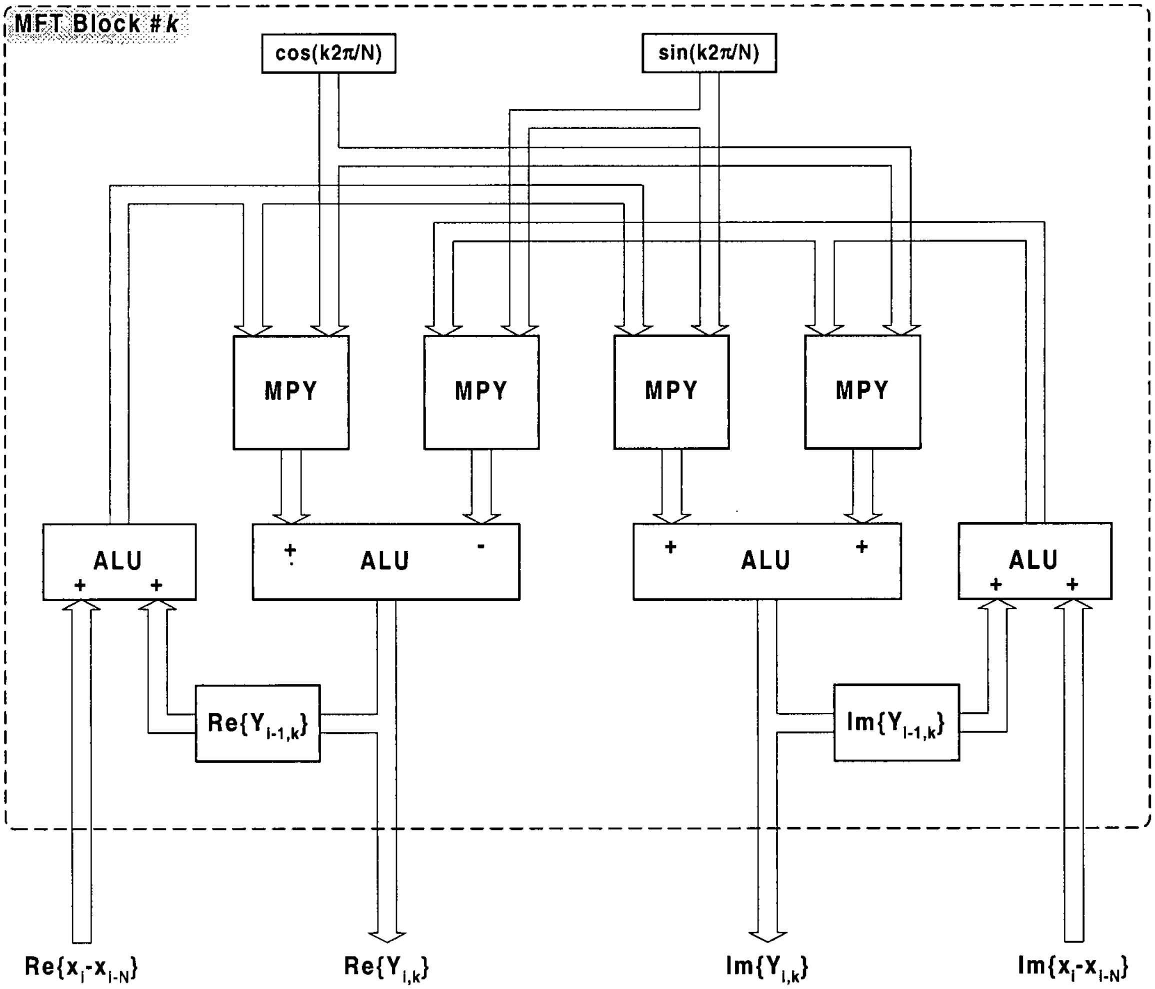


Figure 4 Hardware structure of one MFT block

2.3.3 Example of MFT Usage

To illustrate the usage of the incremental form of the MFT, a frequency shift key (FSK) modulated sinusoidal signal of length *4N* samples is used. Using an analysis window length *N*=100, and two frequencies of 5 cycles/window and 29 cycles/window, the magnitude of the evolving spectrum is shown in Figure 5, when the MFT is incremented by one sample at each analysis stage.

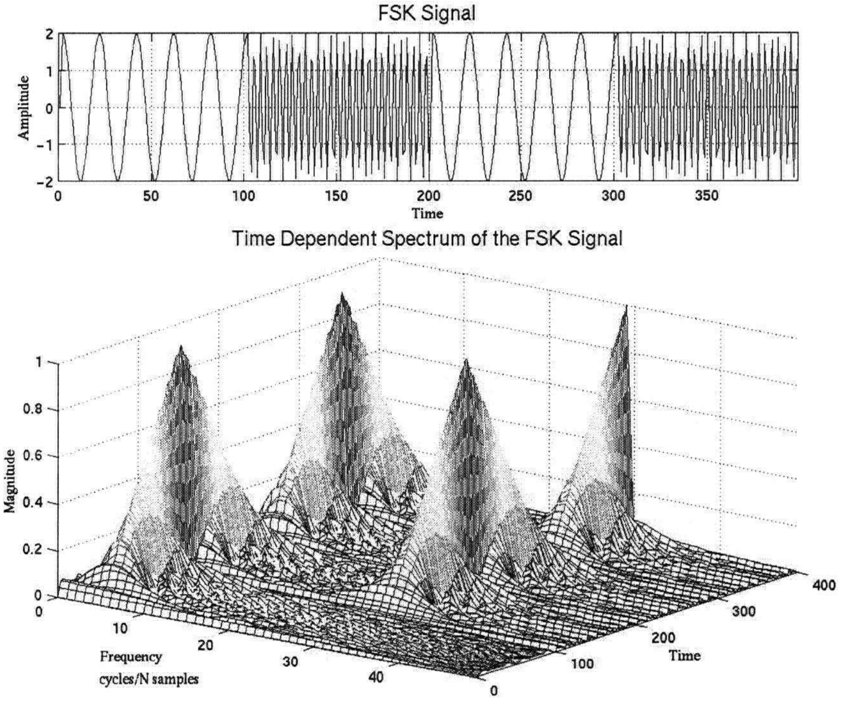
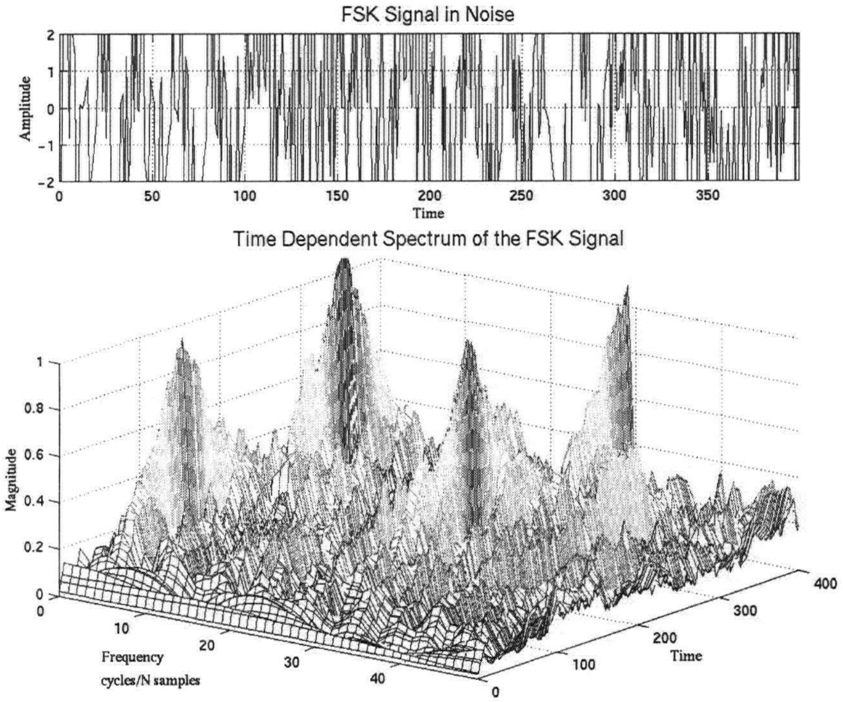


Figure 5 FSK signal analysis using MFT

The MFT begins with the initial conditions of $y_0 = 0$. This is equivalent to having *N* zeros precede the data vector. In Figure 5, note how the energy in the spectrum rises from zero to a maximum in the first *N* samples. Also note how spectral leakage is observed in the first *N*-1 time samples, because the sinusoidal signal does not have an integer number of cycles/window over this time. At time *N*, there is an integer number of cycles/window, so all the energy in the spectrum lies in one bin. For the next *N*-1 samples, leakage occurs again as the window sliding towards to the next frequency component of the signal. The spectral energy of the *5th* frequency bin decays to zero while the spectral energy of the *29th* bin rises to its maximum. This spectrum energy 'swapping' between the two frequency bins is repeated as the window is moving through the two frequency components.

In Figure 6, the same FSK signal is analyzed in the present of noise. The spectrum energy swapping between the two frequency bins is also noticeable, which shows how the MFT can be useful for signal detection in noise environment.


FSK Signal in Noise

Time Dependent Spectrum of the FSK Signal

Figure 6 Signal detection using MFT

2.4 Computing Efficiency of MFT

In this section, a survey on the arithmetic of the MFT is given, followed by a discussion of its efficiency. The MFT is particularly efficient compare to FFT algorithms, when successive DFTs with high overlap ratio are to be computed or when only a few spectrum coefficients are needed. Examples of applications of the MFT to signal processing is also given, here.

2.4.1 Arithmetic of MFT

The previously derived equation for one spectrum component of an *N* samples long MFT at time *i*:

$$y_{i,k} = w^{-k}\left(y_{i-1,k} + x_i - x_{i-N}\right)$$

The twiddle factors (w^{-k}) can be calculated only once and stored in an array before the MFT procedure. This computation is not included in the arithmetic of MFT.

The difference of the sample entering and leaving the window - x_i-x_{i-N} - can be pre-calculated at each time moment and used for the calculation of all spectrum coefficients. In this case, the spectrum of $\mathbf{x}_i$ can be updated from the spectrum of $\mathbf{x}_{i-1}$ using only *N complex multiplies and N+1 complex adds*, if $\mathbf{x}_i$ contains complex-valued data. If $\mathbf{x}_i$ is real, *N/2 complex multiplies and N/2+1 real adds* are needed to obtain the N/2 new spectrum coefficients. Table 2 gives a summary of the number of real operations for these cases when only N_c coefficients are calculated ($N_c \leq N$ for complex data and $N_c \leq N/2$ for real data).

Input data	Real Multiplies	Real Adds	Real Operations
Real	$4N_c$	$3N_c+1$	$7N_c+2$
Complex	$4N_c$	$4N_c+2$	$8N_c+2$

Table 2 Real operations in MFT for N_c coefficients

Note, the number of operations in each case can be reduced with one, if the DC component is calculated, because in that case the twiddle factor equals to 1 ($w^k = 1$ when $k=0$).

2.4.2 Comparison of MFT to FFT algorithms

Consider the case where N point DFTs are used to analyze an M-point complex-valued data record. If the window is shifted by q samples between each DFT application, where $1 \leq q \leq N$, then $\frac{M-N}{q}+1$ DFTs are needed to spectrum analyze the record, in the case of FFT. If the MFT is applied, M MFTs are needed, because the spectrum coefficients have to be calculated in each time samples, irrespectively of the value of q.

Then, when radix-2 FFTs are used:

$$OPS_{FFT} = \left(\frac{M-N}{q}+1\right)[5N\log_2(N)] \tag{31}$$

real operations, while in the case of MFT:

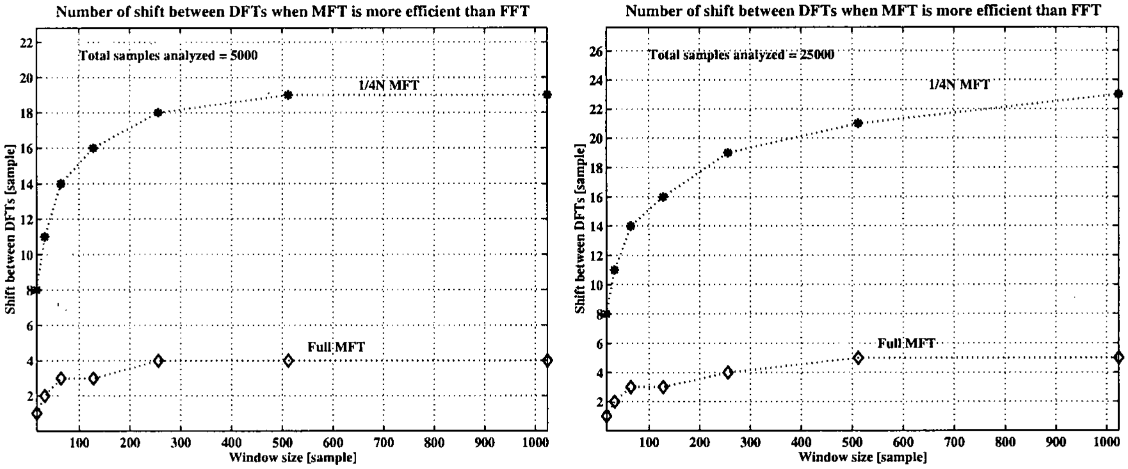
$$OPS_{MFT} = M\left[8N_c + 2\right] \tag{32}$$

real operations are needed to analyze the whole record.

From (31) and (32) the number of shift between DFTs when the MFT is more efficient than the radix-2 FFT can be expressed:

$$q_{MFT} < \frac{(M-N)\left[5N\log_2(N)\right]}{M(8N_c\text{-}1)\text{-}5N\log_2(N)} \tag{33}$$

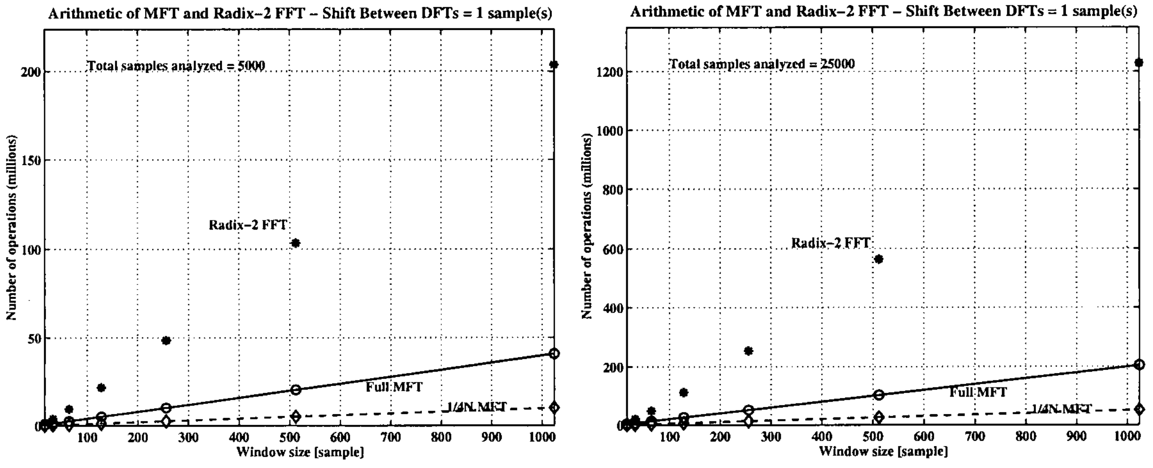
As we can see from (33), q_{MFT} is function of the length of the data record (M), the size of the window (N) and the calculated MFT spectrum coefficients (N_c). In Figure 7, the shift between DFTs when the MFT is more efficient is shown as a function of the window length, with two values of M and N_c:



(a) (b)

Figure 7 Shift between DFTs when the MFT is more efficient

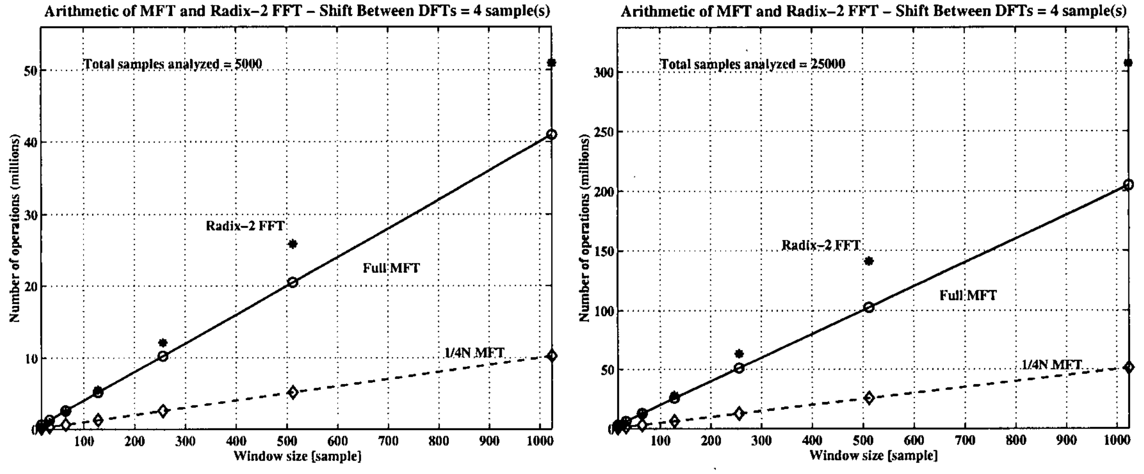
The full MFT is more efficient compared to the radix-2 FFT, if the shift between DFTs is very small ($q_{MFT} \leq 5$), while for the reduced MFT (Nc = $N/4$), the MFT is more efficient even for larger values of shift. Note, if the data record is longer (Figure 7 (b)), the values of q_{MFT} are larger for all window sizes. The computational load for small amount of shifts is illustrated in Figure 8 and 9:



(a)

(b)

Figure 8 Arithmetic of MFT and Radix-2 FFT when $q_{MFT} = 1$



(a)

(b)

Figure 9 Arithmetic of MFT and Radix-2 FFT when $q_{MFT} = 4$

The arithmetic of MFT is linear with the number of the computed spectrum coefficients (N_c) and the length of the data record (M). For a given record size (e.g. Figure 8 (a) and Figure 9 (a)) the MFT arithmetic remains the same, with varying shifts, while the FFT arithmetic drops down considerably as the value of shift gets larger.

In Figure 10 the floating point operation per DFT is shown for radix-2 FFT, mixed-radix FFT, MFT and the direct DFT, when the consecutive windows are overlapped by *N/2* samples (i.e. $q_{MFT} = N/2$). The arithmetic of the mixed-radix FFT was estimated using the Matlab's *fft* and *flops* functions. The radix-2 FFT is very efficient if the DFT length is power of two, while the MFT is more efficient when *N* is a non-composite number (e.g. prime).

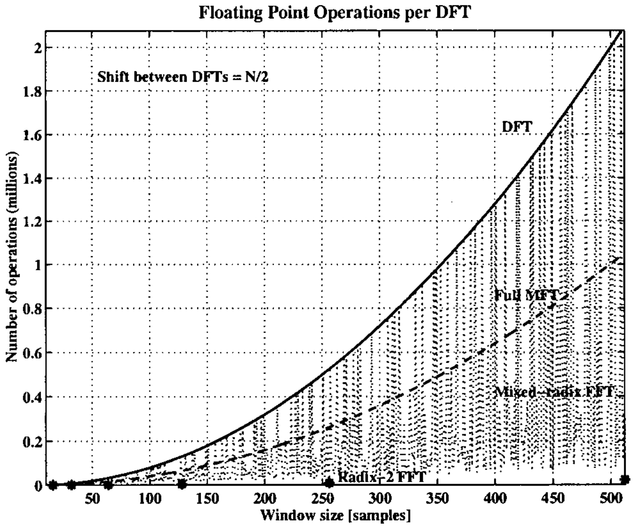


Figure 10 Floating Point Operations of DFT algorithms

When there is no overlap between the two consecutive DFTs (i.e. $q_{MFT} = N$) during the spectrum analysis, the MFT has to be applied *N* consecutive times to obtain the next valid

DFT. In this case, the arithmetic of the MFT becomes the same as the direct DFT in Figure 10, while the arithmetic of the other algorithms remains the same.

As an example of efficiency of the MFT, consider an N-point frequency domain filter with a fast convolution method applied to the complex-valued data record. The filter coefficients are pre-calculated and stored and the filter is applied with a radix-2 FFT (or MFT), an array multiply, then an inverse FFT (or IMFT). When radix-2 FFT is used to obtain the convolution

$$COPS_{FFT} = M\frac{5N_{FFT}}{N}\log_2\left(2^6 N_{FFT}\right), \; where\; N_{FFT} > 2N \tag{34}$$

real operations are needed. In the case of the MFT the number of real operations needed are:

$$COPS_{MFT} = M\left[10N_c + 2\right] \tag{35}$$

The comparison of efficiencies is shown in Figure 11:

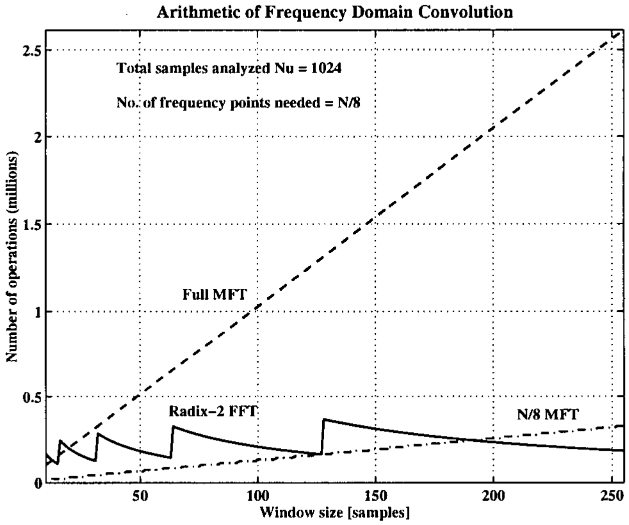


Figure 11 Fast Convolution with MFT and Radix-2 FFT

If the MFT is used to compute all the spectrum coefficients, then the radix-2 FFT is more efficient for most of the DFT lengths. But if only a subset of the spectrum coefficients need to be computed (i.e. sub-band filtering), the MFT/IMFT transformation pair can be more efficient for many values of *N*.

2.4.3 Advantages and Uses of MFT

The computational order of the MFT to recursively calculate the coefficients of an *N*-point DFT is *N*, a factor of log_2N improvement over the FFT. If only a sub-set of the spectrum components are needed, the computing load of the MFT can be further reduced, calculating only the frequency coefficients of interest. The MFT does not rely upon on *N* being power of two to obtain its efficiency, in contrast to standard FFT algorithms. In this way, the MFT can provide more efficient computation of the DFT when any or all of the following conditions apply:

- DFTs are highly overlapped
- only a few Fourier coefficients are needed
- a specific, non-composite DFT length is needed.

Concerning the above properties of the MFT, we can say that it can be useful in different applications of signal processing such as:

- on-line computations in real-time spectral analysis
- on-line signal identification and detection
- speech processing
- radar and sonar processing.

Chapter 3

Overview of SAR Processing

3.1 Introduction

In this chapter the basic geometry of the synthetic aperture radar (SAR) system, the mathematical form of the ideal received signal of a point target and the traditional SAR compression algorithm, the range-Doppler algorithm are introduced.

3.2 Ideal point-target model

Assume, the airplane or satellite carrying the SAR antenna travels across the surface of the earth at a constant velocity (V_r) while transmitting microwave pulses at a given pulse repetition frequency (*PRF*) to the ground with a squint angle Θ as it is shown in Figure 12. The direction of travel of the SAR antenna is called the azimuth direction while the direction of travel of the transmitted pules is referred to as the range direction (azimuth and range directions are perpendicular to each other).

The transmitted pulses travel at the speed of light (*c = 3x10^8 m/s*) which is much faster than the velocity of the antenna. In this way, the antenna can be treated as stationary from the time when it sends out one pulse and receives the reflections from the ground targets. Then the antenna moves to one position to the next azimuth position, sending out another pulse and receiving the back scatter again. Because of the large disparity in time duration of the two directions, azimuth is referred to as the *"slow time"* axis while range as the *"fast time"* axis.

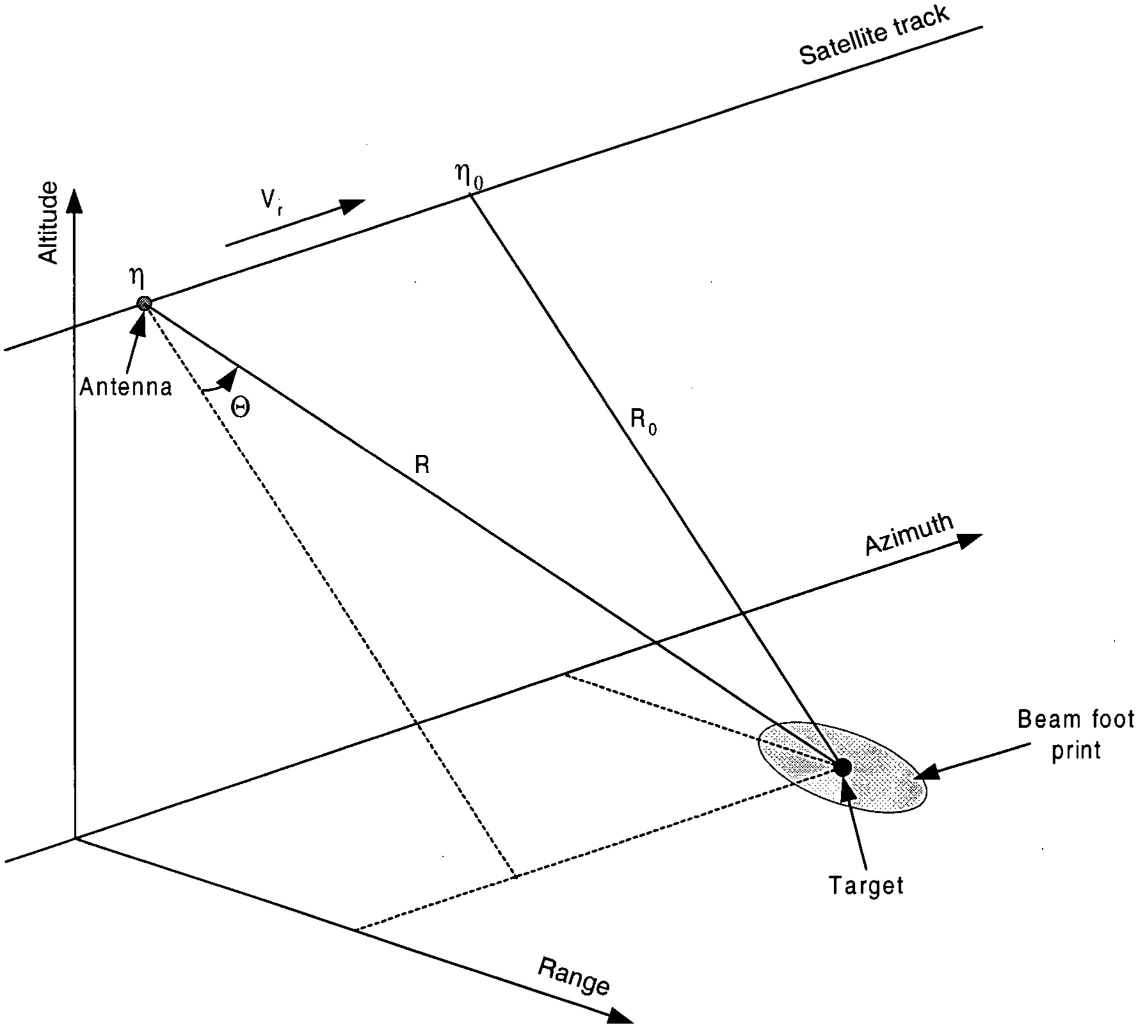


Figure 12 Synthetic aperture radar geometry

Let η represent the slow time variable in azimuth direction. Then at each η there is a fast time variable t corresponds to the signal in the slant range (R) direction. The transmitted pulse is a chirp signal, $\exp\left(j\pi K_r t^2\right)$, and the ideal received SAR signal from a point target can be written as a two dimensional signal [3]:

$$s(t,\eta) = P(t)\ A(\eta)\ \exp\left(j\pi K_r\left(t - \frac{2R(\eta)}{c}\right)^2 - j\pi\frac{4R(\eta)}{\lambda}\right) \tag{36}$$

In (36) $P(t)$ is the pulse envelope in range direction, $A(\eta)$ is the azimuth antenna pattern, K_r is the range FM rate, and λ is the radar wave length. The received signal $s(t,\eta)$ can be separated to the range signal $s_r(t,\eta)$ and the azimuth signal $s_a(\eta)$ as follows:

$$\begin{aligned} s(t,\eta) &= s_r(t,\eta)\times s_a(\eta) \\ s_r(t,\eta) &= P(t)\ \exp\left(j\pi K_r\left(t - \frac{2R(\eta)}{c}\right)^2 \right) \\ s_a(\eta) &= A(\eta)\exp\left(- j\pi\frac{4R(\eta)}{\lambda}\right) \end{aligned} \tag{37}$$

After the chirp travels through the slant range $R(\eta)$ and back, the received signal s_r has a time delay $2R(\eta)/c$. In this case, for the same target, but in a different azimuth position, the time delay of the received chirp is different, causing range cell migration in the data memory.

From the geometry of Figure 12, the closest slant range of the target (R_o) is at azimuth time $\eta = \eta_0$. When the target is at some arbitrary position with respect to the antenna, the slant range R can be expressed as:

$$R(\eta) = \sqrt{R_o^2 + V_r^2 (\eta - \eta_0)^2} \tag{38}$$

Because $R_0 >> V_r(\eta\text{-}\eta_0)$, equation (37) can be approximated by a parabola, expanding the equation in a second order Taylor series around η_0:

$$R(\eta) \approx R_0 + \frac{V_r(\eta - \eta_0)^2}{2R_0} \tag{39}$$

Combining equation (37) and (39) together, the received azimuth signal can be written as:

$$\begin{aligned} s_a(\eta) &= A(\eta)\ \exp\left(-j\pi \frac{4R_0}{\lambda}\right) \exp\left(-j\pi \frac{2V_r^2}{R_0 \lambda} (\eta - \eta_0)^2\right) \\ &= A(\eta)\ \exp\left(-j\pi \frac{4R_0}{\lambda}\right) \exp\left(-j\pi\, K_a (\eta - \eta_0)^2\right) \end{aligned} \tag{40}$$

where the K_a is the azimuth FM rate. Note, that the value of R_0 changes for each range cell, therefore the azimuth FM rate changes also, and this must be taken into account when processing the data from different range locations. Also note, that $s_a(\eta)$ has a constant phase $-4\pi R_0/\lambda$ proportional to R_0. This constant phase must be preserved or recovered after the azimuth compression. It is needed for further SAR processing applications, such as SAR interferometry (InSAR).

The time variable η in the azimuth signal is valid within the exposure period of the target, which is determined by the antenna pattern $A(\eta)$. When the antenna length is L, the foot print width of the antenna beam at the target is $\lambda R_0/L$, so the target exposure time is

$$T_e = \frac{\lambda R_0}{L V_r}$$

(41)

Let η_c represent the azimuth time when the beam center crosses the target. Then, $\eta_c = R_0 tan(\Theta/V_r)$, and the valid interval of η for $s_a(\eta)$ is:

$$\eta_c - \frac{T_e}{2} \;\leq\; \eta \;\leq\; \eta_c + \frac{T_e}{2}$$

(42)

3.3 SAR signal compression

The received SAR data in both range and azimuth can be modeled as the convolution of a linear FM chirp and the ground reflectivity. Using the form of the ideal received signal in equation (36) a matched filter can be derived for each dimension and a pulse compression can be performed on the received data. The pulse compression rearranges the energy received from each ground targets into a single focused pulse. The location of the maximum energy of the pulse corresponds to the location of the target in range and azimuth, while the strength of the pulse represents the reflectivity of the target.

The Range/Doppler (RD) algorithm is a traditional, highly accurate and efficient method for compressing SAR data. It consist of the following major stages [3]:

- range FFT,
- range matched filter multiplication,
- range IFFT,
- azimuth FFT,
- range cell migration correction (RCMC),
- azimuth matched filter multiplication and
- azimuth IFFT.

In the RD algorithm, the range and the azimuth signal are compressed separately using different matched filters. In either case, the matched filtering can be implemented via time domain convolution or frequency domain multiplication. In the rest of this section, the pulse compression is introduced through the frequency domain azimuth compression.

The azimuth signal in (40) can be simplified without loss of generality by ignoring the phase term $exp(-j\pi 4R_0/\lambda)$ and the antenna pattern $A(\eta)$:

$$s_a(\eta) = \exp\left(-j\pi K_a(\eta - \eta_0)^2\right)$$

The spectrum of the signal in equation (43):

$$S_a(f) = rect\left(\frac{f - F_{dc}}{K_a T_e}\right)\exp\left(j\pi\left(\frac{f^2}{K_a} - 2f\eta_0\right)\right) \tag{43}$$

Where F_{dc} is the Doppler centroid frequency and $F_{dc} = -K_a(\eta_c - \eta_0)$. The Doppler bandwidth of the azimuth signal $BW = T_e K_a$.

The matched filter in the azimuth frequency (Doppler) domain is the complex conjugate of $S_a(f)$:

$$M(f) = S_a^*(f)_{\eta_0=0}$$

(44)

The frequency domain compressed signal is the product of the spectrum of the matched filter and the spectrum of the azimuth signal:

$$\begin{aligned} S_c(f) &= M(f) \cdot S(f) \\ &= rect\left(\frac{f - F_{dc}}{K_a T_e}\right) \exp(-j2\pi f \eta_0) \end{aligned}$$

(45)

The compressed signal in the time domain is the inverse Fourier transform of $S_c(f)$:

$$s_c(\eta) = F\{S_c(f)\} = K_a T_e \exp(j2\pi F_{dc}(\eta - \eta_0)) \, \text{sinc}(K_a T_e(\eta - \eta_0))$$

(46)

In (48), the compressed peak is at the point of the target's closest approach (η_0). This compressed peak location can be changed to other position, such as the target's starting time or Doppler centroid location, by changing the format of the matched filter.

In the following two chapters, survey on the applicability of the Momentary Fourier Transform to SAR signal processing algorithms is given. Chapter 4 shows how the MFT can be applied to the SPECAN SAR processing algorithm, while in Chapter 5 it is shown what advantages the MFT offers when it is used for burst-mode data processing.

Chapter 4

Application of MFT to SPECAN SAR Processing Algorithm

4.1 Introduction

As it was mentioned in the previous chapter, SAR signal compression in range and azimuth can be accomplished by cross-correlation in the time domain using time domain convolution, or in the frequency domain, using the fast-convolution variant Range-Doppler method. Alternately, advantage can be made of the linear FM structure of signals by replacing the cross-correlation operation with a frequency filtering operation performed by a DFT. This method is called SPECtral ANalysis (SPECAN) [11]. In this chapter after the theory of the SPECAN algorithm discussion on the application of MFT to azimuth SPECAN processing is given.

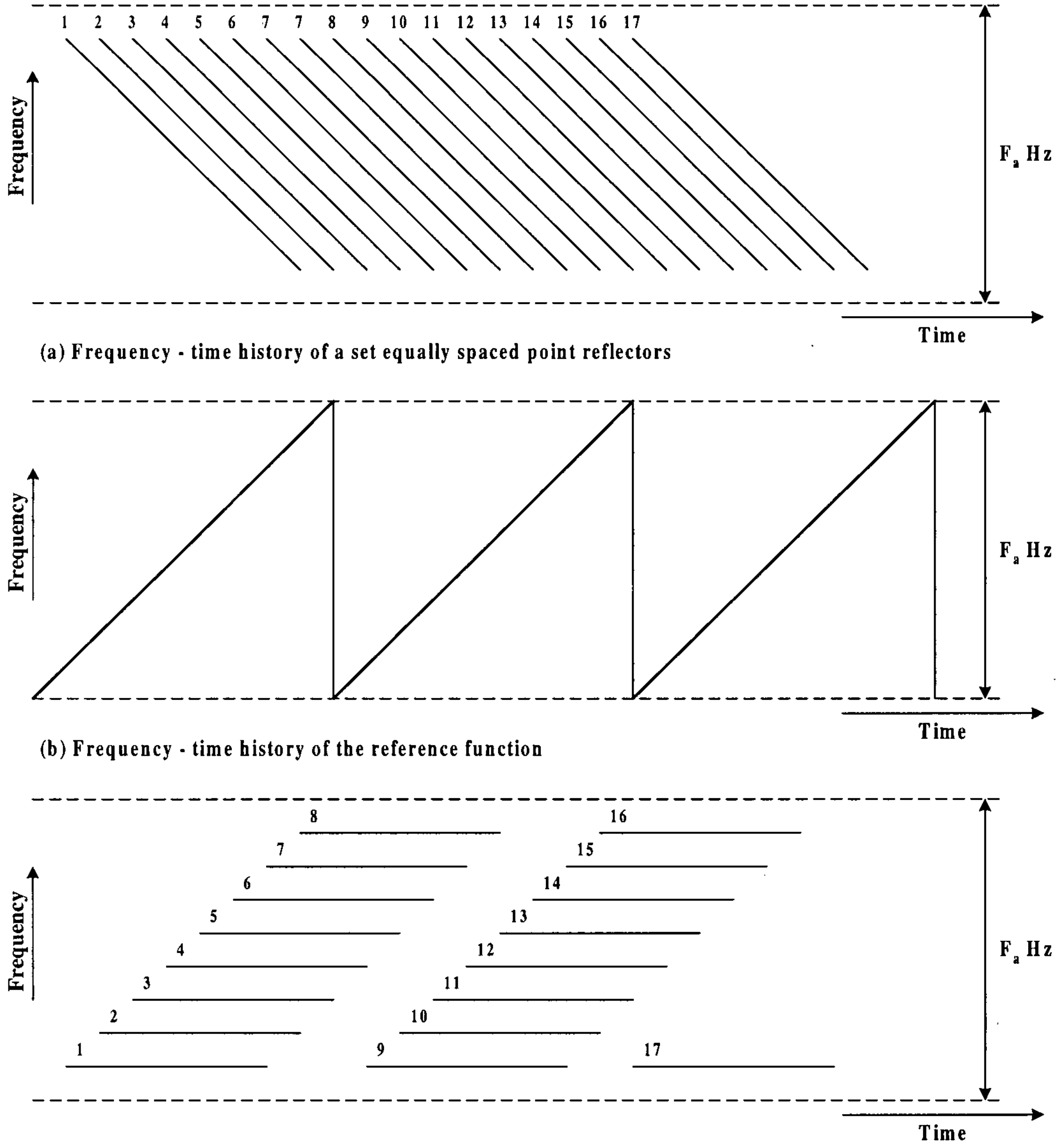
4.2 The SPECAN Algorithm

The SPECAN algorithm consists two major computational steps:

- Deramping

- DFT extraction.

Deramping is the operation of multiplying a linear FM signal with a complex conjugate reference signal with the same FM rate, but opposite FM slope. The deramping of a signal containing multiple targets is shown in Figure 13.



(a) Frequency - time history of a set equally spaced point reflectors

(b) Frequency - time history of the reference function

(c) Frequency - time history of the product function when the signal in *(a)* is multiplied by the reference function *(b)*

Figure 13 Deramping of multiple targets

After deramping, the targets whose zero Doppler frequency is located within one reference function cycle will have frequencies ranging from $-F_a/2$ to $F_a/2$, where F_a is the azimuth sampling frequency. These targets will constitute one processing region. The formation of parallelogram shaped regions from the deramped targets is shown in Figure 13 (c).

Note that each incremental step in the time direction in Figure 13 (a) results an incremental step in the frequency direction in (c), and that the frequency continuity is reset by F_a Hz (i.e. $f = 0$ and $f = F_a$ are connected).

Consecutive processing regions are separated by lines of slope of the azimuth FM rate (K_a) and constitute a parallelogram shaped area. Figure 14 shows one processing region in more detail. The parameters defined on the figure are calculated as follows:

- Azimuth FM rate: $K_a = \frac{2V_r^2}{\lambda R_0}$ [Hz/s]
- One cycle of the reference function: $M = \frac{F_a^2}{K_a}$ [samples]
- The processed region of the total Doppler spectrum: $M_p = (1-\beta)M$, where parameter β denotes the guard band.
- Velocity of a sub-satellite point: V_r [m/s]
- Wavelength: λ [m]
- Closest slant range: R_0 [m]
- Sampling frequency of the azimuth signal: F_a [Hz]

Each deramped target in the parallelogram has a unique frequency ranging from $-F_a/2$ to $+F_a/2$. It is this unique frequency which defines the position of each target with respect to other targets in the same parallelogram.

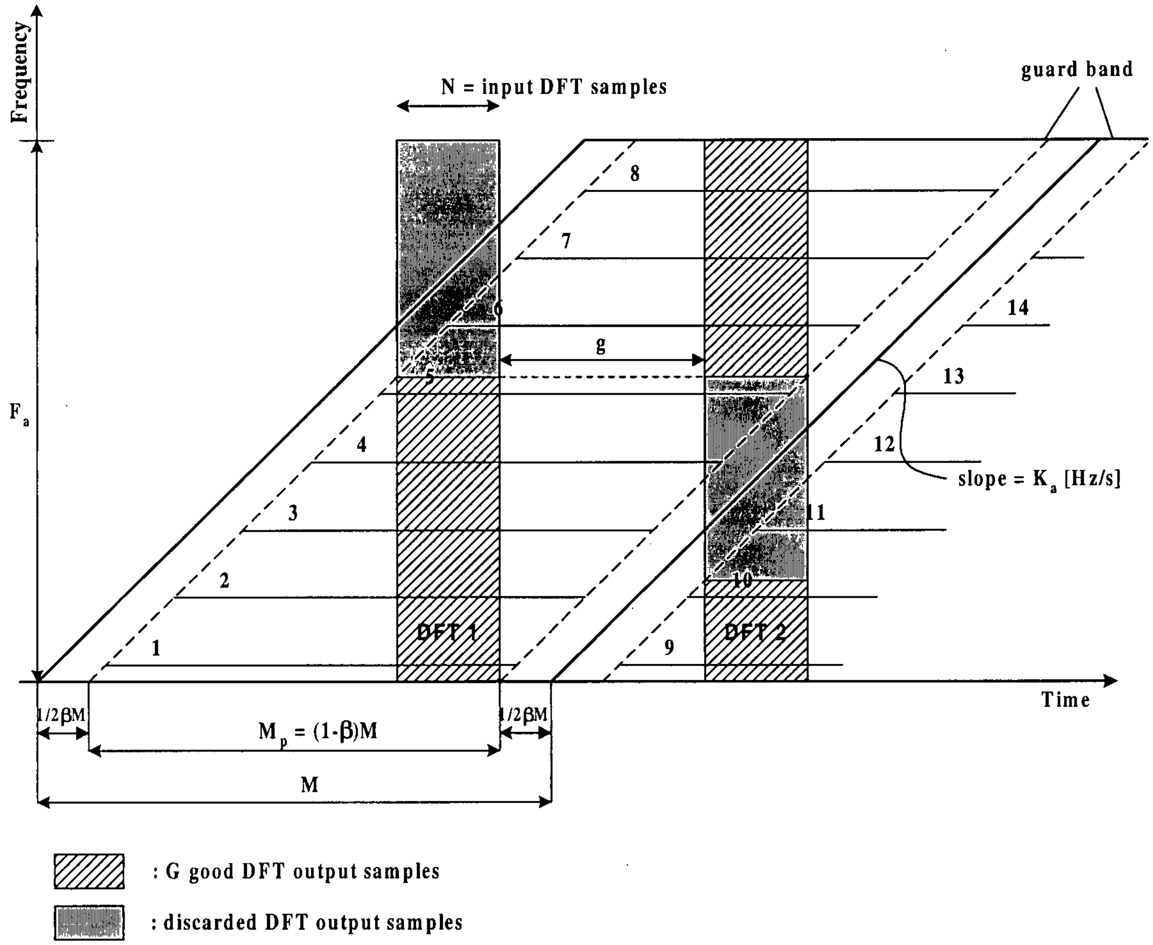


Figure 14 Processing regions and the placement of successive DFT blocks in single look case

The parts of the energy of any target which originate near the ends of each trajectory (i.e. near the sloped lines) has poor SNR because they arise from the low gain part of the azimuth beam profile and because of the relatively high presence of aliased energy there.

The guard band is represented by dashed lines in Figure 14. The total width of the guard band in the time direction is β times the distance between the sloped lines. This means that only $(1\text{-}\beta)$ fraction of the total available Doppler spectrum of each target trajectory is processed.

The next step in the processing is to separate the targets into different energy cells with regard to their position in the time domain. This is done by performing short length DFTs across the deramped data. The placement of the DFTs is also shown in Figure 13, where the DFT length is N samples. The location of the first DFT block is arbitrary, but for the sake of illustration it is placed so that the last sample corresponds to the bottom right corner of the processing region. The first DFT rectangle is divided into two parts by a horizontal line where the left-hand side of the rectangle touches the right side of the guard band. The upper section of the rectangle contains invalid output samples which must be discarded because the targets in that region are not fully exposed, while output points corresponding to the lower section of the rectangle are kept as the good DFT output points.

If $N = \alpha M$, then the unused portion of the available time axis is $(1\text{-}\alpha\text{-}\beta)\, M$ samples long and the height of the valid part of the rectangle is

$$G = (1\text{-}\alpha\text{-}\beta)\, N \tag{28}$$

DFT output samples. G is the number of good points retained from the DFT operation.

The next DFT must be applied so that the frequency of the lowest frequency cell corresponding to the good output region is exactly one cell higher than the highest frequency cell of the current DFT. This placement is shown in Figure 14, as well. The gap between successive DFTs in the input time domain:

$$g = (1-2\alpha-\beta)M \tag{29}$$

The DFT length is governed by the desired azimuth resolution ρ_a, by the relation:

$$N = \frac{0.89\, V_r F_a}{\sigma\, \rho_a K_a} = \frac{0.89\, F_a \lambda\, R_0}{\sigma\, \rho_a 2\, V_r} \tag{30}$$

where σ is the weighting factor used in the application of DFT such that σN is the effective number of samples used in the DFT input. From (30) it is seen that the DFT length (N) is directly proportional to the range (R_0) and inversely proportional to the resolution (ρ_a), while the variables V_r, F_a and K_a are defined by the SAR system. The azimuth FM rate strongly depends upon the range while V_r depends weekly upon R for satellite systems and constant for airborne systems. The azimuth resolution can be expressed from (30):

$$\rho_a = \frac{0.89\, V_r F_a}{\sigma\, N K_a} = \frac{0.89\, F_a \lambda\, R_0}{\sigma\, N 2\, V_r} \tag{31}$$

Table 3 gives available azimuth resolutions for various DFT lengths, with the following C-band SAR satellite parameters: K_a = *2100 Hz/s*, F_a = *1650 Hz*, V_r = *6800 m/s*, σ = *0.68*.

DFT length - N	Resolution - ρ_a
128	45 m
256	23 m
512	11 m

Table 3 Resolution versus DFT length in SPECAN for C-band satellite SAR

4.3 Multi look processing in SPECAN

The processing scheme of Figure 14 is a single look processing because the azimuth compression operation produces only one output point for each azimuth location. A multi look processing scheme is introduced as follows.

The large gaps between DFTs in Figure 14 shows that much of the input data is not used, which is indicative of the excess azimuth bandwidth available when this length of DFT is used. This excess bandwidth can be used to generate multiple looks.

Extra looks can be generated by dividing the *G* good output points from a single DFT into N_l equal groups, and assigning each group to a different look number. N_l is the number of looks. The next DFT is then placed so that the beginning of its good output points are contiguous with the end of the first look of the first DFT. Such a division into looks and placement of the second DFT is illustrated in Figure 15 for the four-look case. In this case the target is fully exposed in four consecutive DFTs, and the fifth DFT would have the same location as the second DFT in Figure 14.

Using the good points, obtained from each DFT output (28), the number of good output points per look is

$$L=INT\left\{\frac{G}{N_l}\right\}=INT\left\{\frac{(1-\alpha-\beta)N}{N_l}\right\}$$

(32)

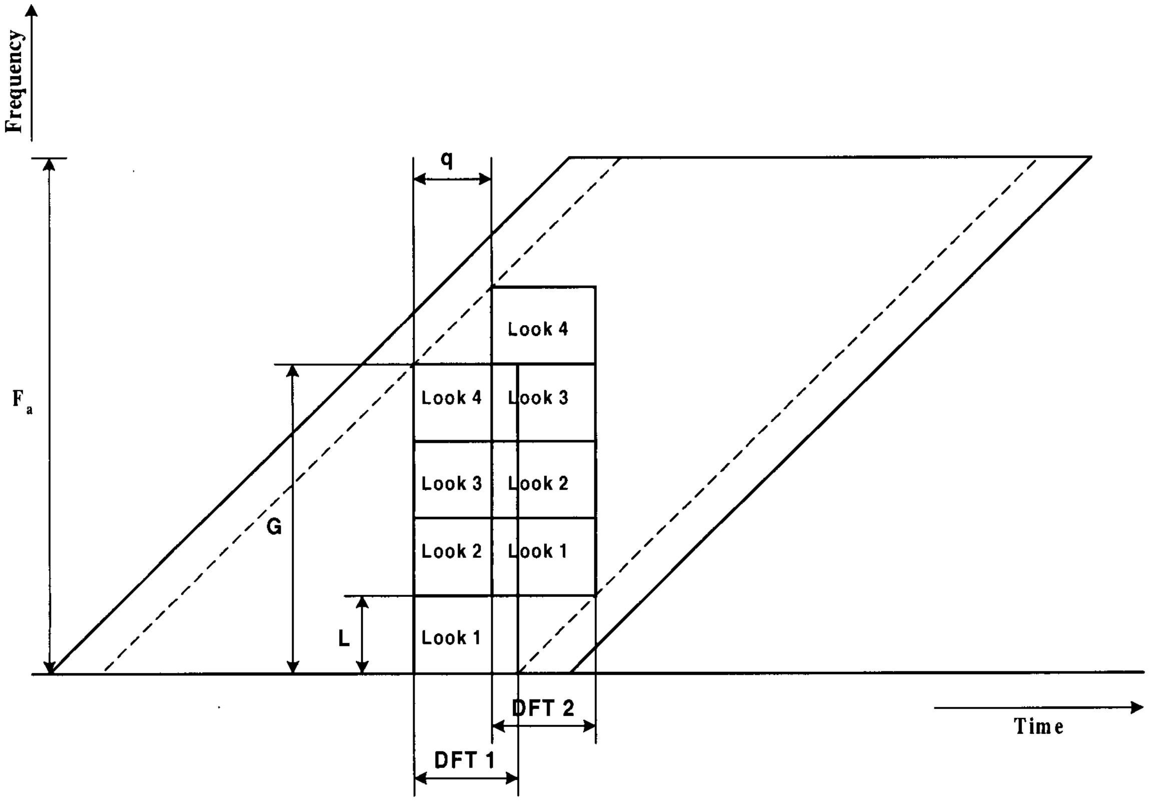


Figure 15 Division of the good output samples into looks and the location of the DFT operations in multi-look processing

Note that G/N_l is not in general an integer so that usually $LN_l < G$ and that a small fraction of the good output points are not used. L must be an integer so that the looks fit together evenly at look summation time for each successive DFT.

The shift between successive DFTs is:

$$q = \frac{L F_a^2}{N K_a} = \frac{R_0 \lambda}{2 \rho_a} \left(\frac{F_a (1 - \beta)}{V_r} - \frac{0.89}{\sigma \rho_a} \right) \tag{33}$$

In general q is not an integer, and the nearest integer must be chosen in order to decide where to place the second DFT.

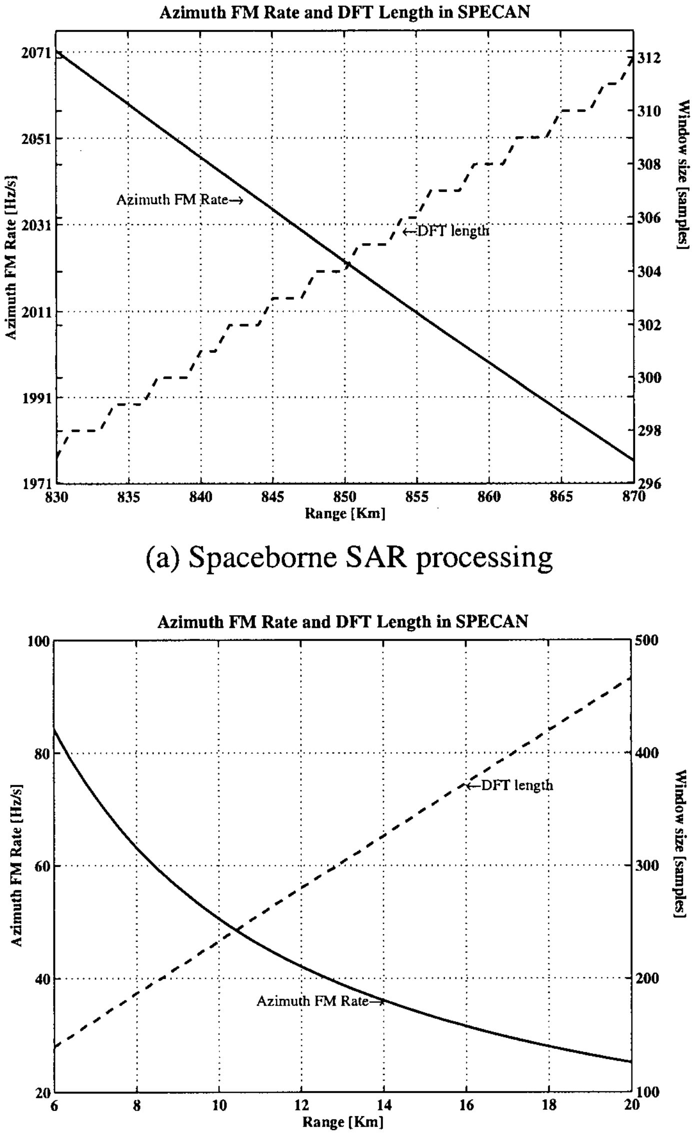
For complete efficient data utilization $q \approx \sigma N$, so the number of looks that are normally taken can be expressed as follows:

$$N_l = \frac{F_a^2}{\sigma N K_a} (1 - \beta) - \frac{1}{\sigma} = \frac{F_a \rho_a}{0.89 V} (1 - \beta) - \frac{1}{\sigma} \approx \frac{F_a \rho_a}{V_r} \tag{34}$$

Note that N_l is linearly proportional to ρ_a and does not depend on R_0.

4.4 The SPECAN Algorithm Using the MFT

The azimuth FM rate of the received SAR signal is inversely proportional to range, so it changes as the range varies in each cell. In order to keep the resolution and output sampling rate constant across the swath, there is a need to choose different DFT lengths which vary with one or a few samples at a time. The affect of the varying range on the FM rate and the required DFT length for spaceborne and airborne case are shown in Figure 16.



(a) Spaceborne SAR processing

(b) Airborne SAR processing

Figure 16 Azimuth FM rate and the DFT length with varying range

The radar parameters for the two cases are given, in Table 4. Note that in the spaceborne case there is a need for only 16 different DFT sizes to keep the resolution constant through the whole swath, while in the airborne case a wide range of window length is needed.

Radar parameter	Spaceborne SAR	Airborne SAR	Units
Velocity (V_r)	7000	120	[m/s]
Wavelength (λ)	0.057	0.057	[m]
Weighting parameter (σ)	0.68	0.68	-
Guard band (β)	0.15	0.15	-
Slant range (R)	830 - 870	6 - 20	[km]
Sampling frequency (F_a)	1680	300	[Hz]
Number of looks (N_l)	6	10	-
Azimuth resolution (ρ_a)	25	4	[m]

Table 4 Spaceborne and airborne SAR parameters for SPECAN arithmetic calculation

The Radix-2 FFT algorithm can be only used when the DFT length is a power of two. In other cases of window length a mixed-radix FFT algorithm is used to achieve efficiency only for highly composite N. It means for each different DFT a different FFT algorithm should be implemented within the SPECAN processor, which makes the DSP architecture rather complex when many lengths of FFT are needed.

In contrast to FFT algorithms, the structure and the efficiency of the MFT does not rely upon on the size of the DFT. The same simple algorithm can be used to calculate all of the necessary DFTs during the azimuth compression. The number of real operations of MFT to process an M samples long region, when all of the spectrum coefficients are computed ($N_c = N$) is given in equation (32).

During the DFT extraction only a portion of the spectrum coefficients - good output samples (G) - are used at the same time to obtain the compressed data. Although the amount of these spectrum components remains the same through the processing, but their position changes with the DFTs. So, the conventional form of the reduced-MFT algorithm cannot be used in this case. The sub-band of the calculated spectrum coefficients has to change its positions in the frequency domain in phase with position of the good output samples. The arithmetic of the required reduced-MFT algorithm is introduced below.

For the first MFT (DFT 1 in Figure 13) all of the good output samples are computed for the first time so the arithmetic of the first MFT is:

$$MFTOPS_{DFT\,1} = N\left[8G+2\right] \tag{35}$$

For the next MFT (DFT 2 in Figure 13) the position of the sub-band of the good output points shifted towards to the higher frequencies with L samples (the good output points in a look). Thus, there are L new frequency components to calculate beside the (N_l-L) continuously calculated ones. The number of real operations needed for the L new coefficients of the second DFT are:

$$MFTOPS_{DFT\,2\,NEW} = N\left[8L+2\right] \tag{36}$$

The arithmetic of the previously computed spectrum coefficients of the second DFT is:

$$MFTOPS_{DFT\,2\,OLD} = q\left[8\left(N_l-1\right)L+2\right]$$

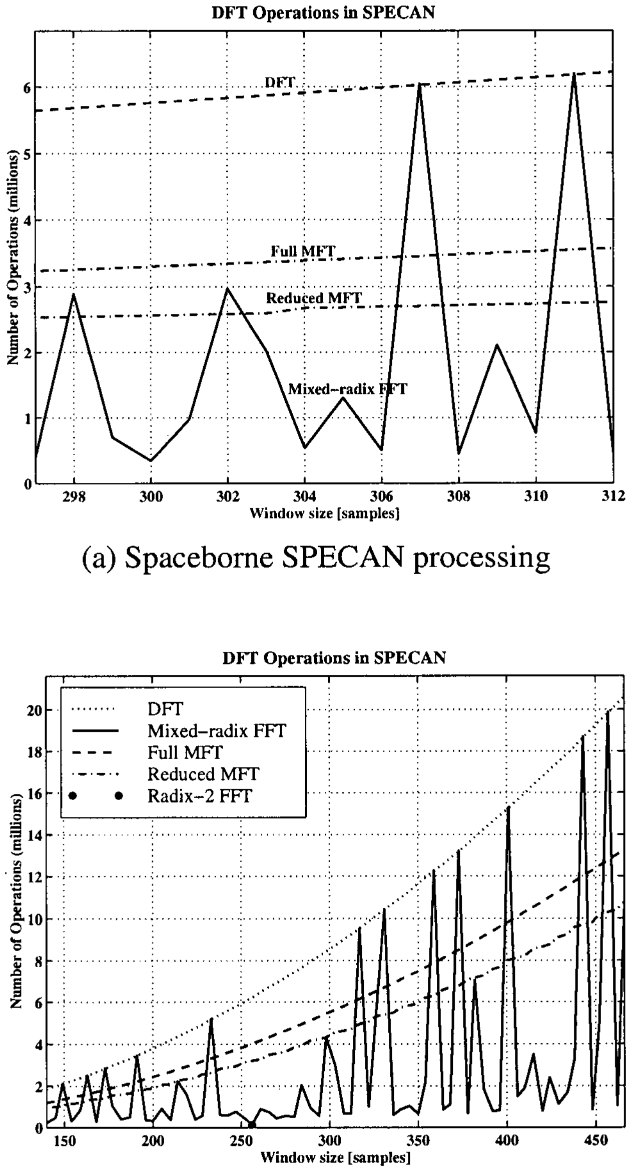
(37)

The shifting of the interval of the good output points in the frequency domain continues during the whole azimuth processing, which means that the 'new' and the 'old' spectrum coefficients have to be computed $\left(\frac{M-N}{q}+1\right)$ times through the whole region. Using the previously introduced equations

$$MFTOPS = MFTOPS_{DFT1} + \left(\frac{M-N}{q}+1\right)\left(MFTOPS_{DFT2NEW} + MFTOPS_{DFT2OLD}\right)$$
$$= N\left[8G+2\right]+\left(\frac{M-N}{q}+1\right)N\left[8L+2\right]+\left(\frac{M-N}{q}+1\right)q\left[8\left(N_l-1\right)L+2\right]$$

(38)

real operations are needed to process the whole processing region with the reduced MFT. Although equation (38) looks rather complex, the implementation of the above described reduced MFT algorithm is the same as the full MFT algorithm, except the timing and synchronization of the sub-band of the spectrum coefficients. Figure 17 shows the number of operations of the SPECAN azimuth compression for spaceborne and airborne cases.



(a) Spaceborne SPECAN processing

(b) Airborne SPECAN processing

Figure 17 Arithmetic of SPECAN azimuth compression with different DFT algorithms

The DFT algorithms used to obtain the DFT output samples are the direct DFT algorithm, the full and reduced MFT, the mixed-radix and radix-2 FFT. Although the radix-2 FFT algorithm is the most efficient, it can be used only once during the process, when the DFT length is 256 samples (Figure 17 (b)). As the window length gets larger (K_a smaller) the size of the processing region also gets larger, thus more operation is needed to process one range cell. The envelope of the plots in Figure 17 (a) and (b) represents the tendency of the growing arithmetic of SPECAN.

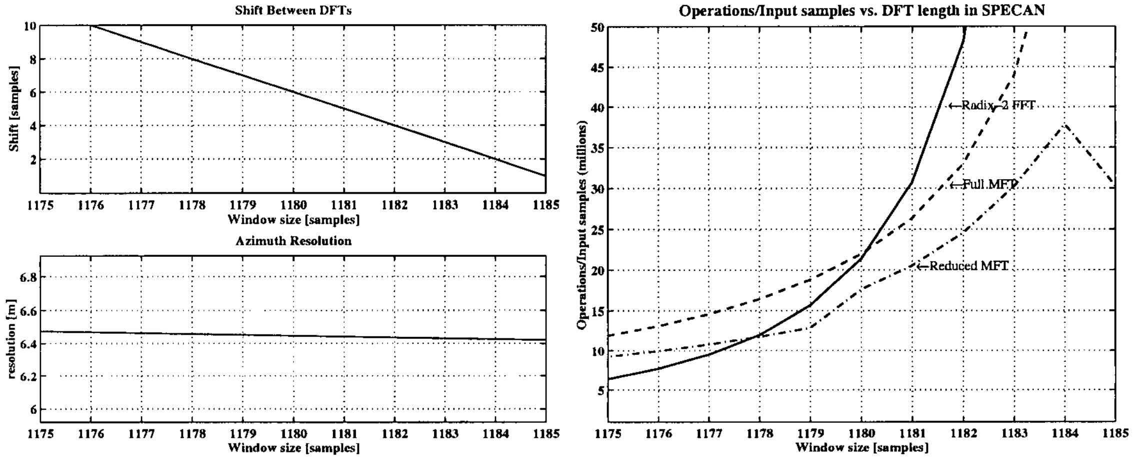
Note that the arithmetic of the full and reduced MFT is more uniform in contrast with the arithmetic of the mixed-radix FFT algorithms. Also note, that the arithmetic of the mixed-radix algorithm is equal to the direct DFT if the window length is a non-composite number (i.e. prime). In Table 5 the ratio between the maximum and minimum of real operation of the reduced-MFT and mixed-radix FFT are given, followed by the ratio between the average operation of reduced and full-MFT and mixed-radix FFT of the full swath.

Ratio of flops	Spaceborne SAR	Airborne SAR
$\frac{\text{Max. of reduced MFT}}{\text{Min. of reduced MFT}}$	1.09	11.15
$\frac{\text{Max. of mixed radix FFT}}{\text{Min. of mixed radix FFT}}$	18.00	141.25
$\frac{\text{reduced MFT per swath}}{\text{mixed radix FFT per swath}}$	1.82	1.74
$\frac{\text{full MFT per swath}}{\text{mixed radix FFT per swath}}$	1.42	2.17

Table 5 Reduced and full MFT versus mixed-radix FFT

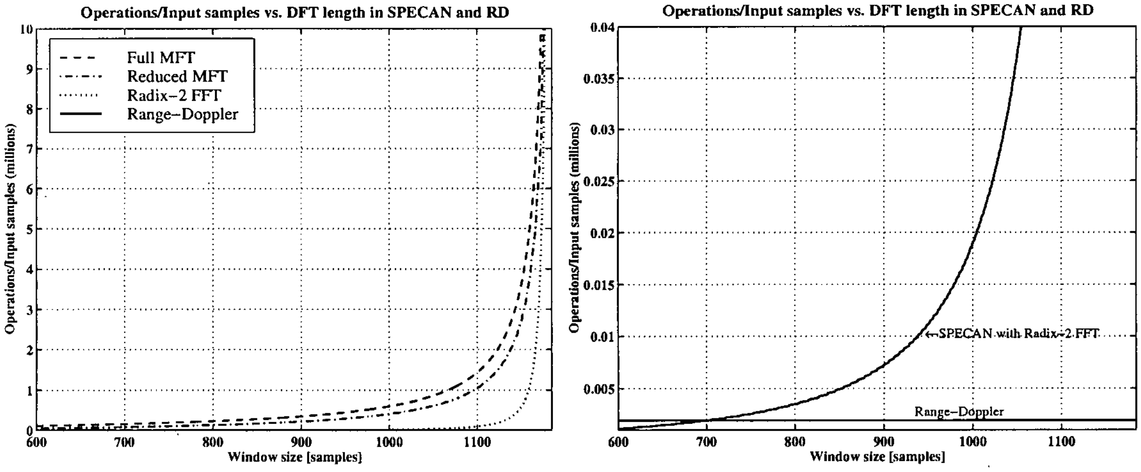
From Table 5 we can see that in the spaceborne case there is less than 10 % difference between the minimum and maximum of the reduced-MFT flops, while the maximum of the real operations of the mixed-radix FFT more than 18 times bigger than the minimum. In the airborne case, the azimuth FM rate changes more dramatically, thus the difference between the minimum and maximum flops are much larger. When the MFT is applied to SPECAN there is a factor of 11 difference between the maximum and minimum arithmetic, while in the case of the FFT the maximum of flops is 141 times bigger than the minimum. This large difference in the processing arithmetic makes the timing of the data flow in SPECAN rather difficult when FFT algorithms are implemented. Note, the results in Table 5 are strongly depend on the azimuth resolution and on the width of the swath.

In the SPECAN algorithm, the resolution is inversely related to the DFT length, thus larger DFTs are needed to obtain finer resolution. As the transformation length is getting wider, the interval of the good output points will shorten, therefore more DFT blocks with higher overlap ratio will be needed to cover the processing region (Figure 14). In this case, the SPECAN algorithm requires more computation to obtain the azimuth compression. Figure 18 and 19 illustrate the real operations per input samples of the traditional Range-Doppler (RD) algorithm and the SPECAN algorithm with the MFT and radix-2 FFT algorithms for a spaceborne SAR system. The used system parameters for the analysis are the same as in Table 4, except the number of looks, $N_l = 1$ in this case.



(a) (b)

Figure 18 The arithmetic of SPECAN when the MFT is more efficient



(a) (b)

Figure 19 The arithmetic of the Range-Doppler algorithm and the SPECAN algorithm using the MFT and FFT algorithms

As it was shown in Section 2.4.2, the MFT algorithm gets more efficient as the Radix-2 FFT when the shift between successive DFTs are not bigger than five samples. Small amount of shift between the DFTs and the corresponding resolution are shown, in Figure

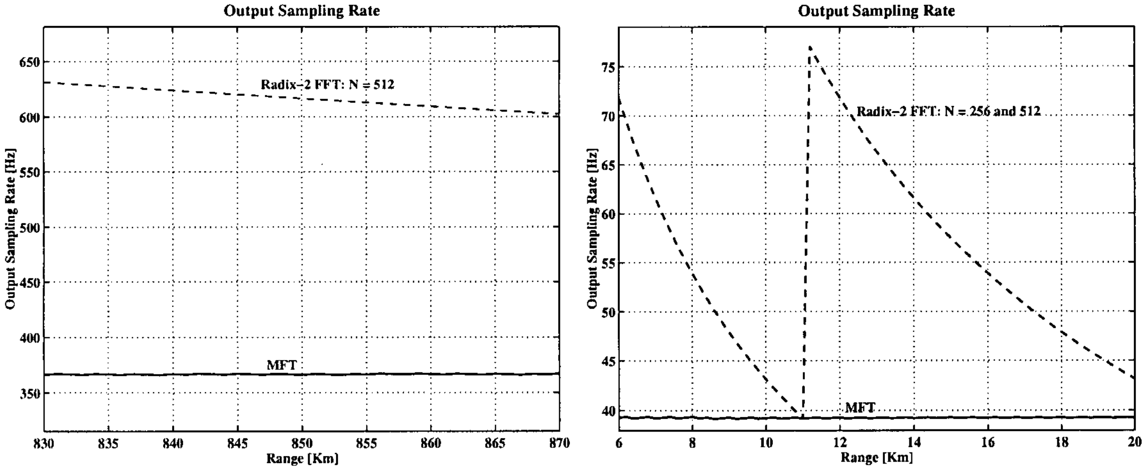
18 (a). The shift between DFTs is smaller than six samples only for large window lengths (1181, 1182, 1183, 1184, 1185 samples), close to the maximum ($N_{max} = (1\text{-}\beta)M = 1186$). Thus the MFT algorithms are more efficient than the Radix-2 FFT algorithm only for very fine resolutions. In Figure 18 (b) the real operations per input samples of the SPECAN algorithm with the full-MFT, reduced-MFT and radix-2 FFT are shown.

Figure 19 (a) illustrates the arithmetic of the RD algorithm and the SPECAN with different DFT implementations. As the DTF length gets larger, thus the azimuth resolution gets finer (i.e. $N > 700$ *samples*, $\rho_a \leq 10.85$ *m*) the RD algorithm gets more efficient, even if the Radix-2 FFT is applied in SPECAN (Figure 19 (b)). Thus, when fine resolution is needed, the RD algorithm is used for SAR signal compression. From Figure 18 (b) and 19 (b) it can be seen that the FFT algorithms are more efficient than the full- or reduced-MFT for all transformation lengths usually used in SPECAN, thus the MFT does not improve significantly the computational efficiency of the SPECAN algorithm.

Beside the complexity and computational efficiency, another important issue in the SPECAN algorithm to keep the output sampling rate constant. In other words, targets which are T seconds apart in azimuth input time must appear T seconds apart in the output data. It was shown in [11] that the azimuth output sample rate is

$$F_{Out} = \frac{K_a N}{F_a} \text{Hz} \tag{40}$$

The output sampling rate strongly depends on the azimuth FM rate, so when K_a changes trough the swath, there is also a need for slowly varying DFT length to keep the output sampling rate constant. Figure 20 shows F_{out} as the function of range, when MFT and radix-2 FFT is applied in the SPECAN algorithm for spaceborne and airborne systems. Note when the MFT algorithm is applied, the output sampling rate is more uniform for both cases. During the application of the radix-2 FFT only two transformation lengths - 256 and 512 - can be used for the airborne case, while only one, $N = 512$ can be used for the spaceborne case. This is the reason of the large migration of the output sampling rate when the radix-2 FFT is applied to the SPECAN.



(a) Spaceborne case

(b) Airborne case

Figure 20 The output sampling rate of the SPECAN algorithm

In this chapter, the applicability of the MFT to SPECAN SAR processing algorithm has been investigated. Although, the MFT does not improve the computational efficiency of the SPECAN algorithm, it has the following advantages over the radix-2 and mixed-radix FFT when they are applied to SPECAN:

- The MFT has consistent computing load as the DFT length changes.
- It is easier to implement the MFT algorithm for variable window length. The architecture of a SPECAN processor using MFT is less complex, because the same MFT algorithm can be used for the different window lengths.
- The full radar resolution can be achieved, because the full Doppler spectrum of the targets can be used for the compression by using high-overlapped DFTs.
- The output sampling rate of SPECAN is more uniform.

Chapter 5

Application of MFT to Burst-mode SAR Data Processing

5.1 Introduction

Burst-mode operation is used in SAR systems, such as RADARSAT, to image wide swaths, to save power or to save data link bandwidth. In this operational mode the received data has segmented frequency-time energy in its Doppler history. There are several algorithms for burst-mode data processing: one of them is the Short IFFT (SIFFT) algorithm which was proposed by Dr. Frank Wong at MDA [17]. In this chapter, after the introduction of the burst-mode data and the SIFFT algorithm properties, a survey on the efficiency of the SIFFT and the applicability of the MFT to the SIFFT algorithm is given.

5.2 Burst-mode SAR processing

Burst-mode is commonly used in SAR systems in ScanSAR mode, where the beam is switched between two or more swaths to maximize the imaged swath width. A 2-beam ScanSAR mode is illustrated Figure 21.

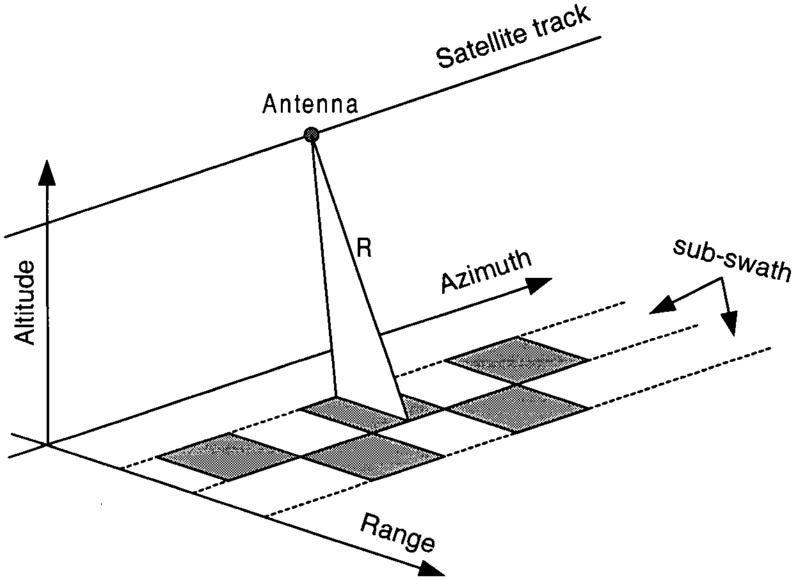


Figure 21 Burst-mode operation in 2-beam ScanSAR case

In this operation mode, the radar beam scans through one sub-swath for a certain time interval, and switches to the next one. After scanning through the second sub-swath, the radar switches back to the first one to start the next burst cycle. The burst cycle has to be short enough to make sure each target is fully exposed in at least one burst.

The data from one sub-swath have to be processed separately from other sub-swaths, because the radar beam covers a different ground area in different sub-swaths. In the azimuth direction, the data is segmented into discrete bursts (shaded area in Figure 21), while in range the signal of any sub-swath is continuous, thus the data is not acquired in discrete range bursts.

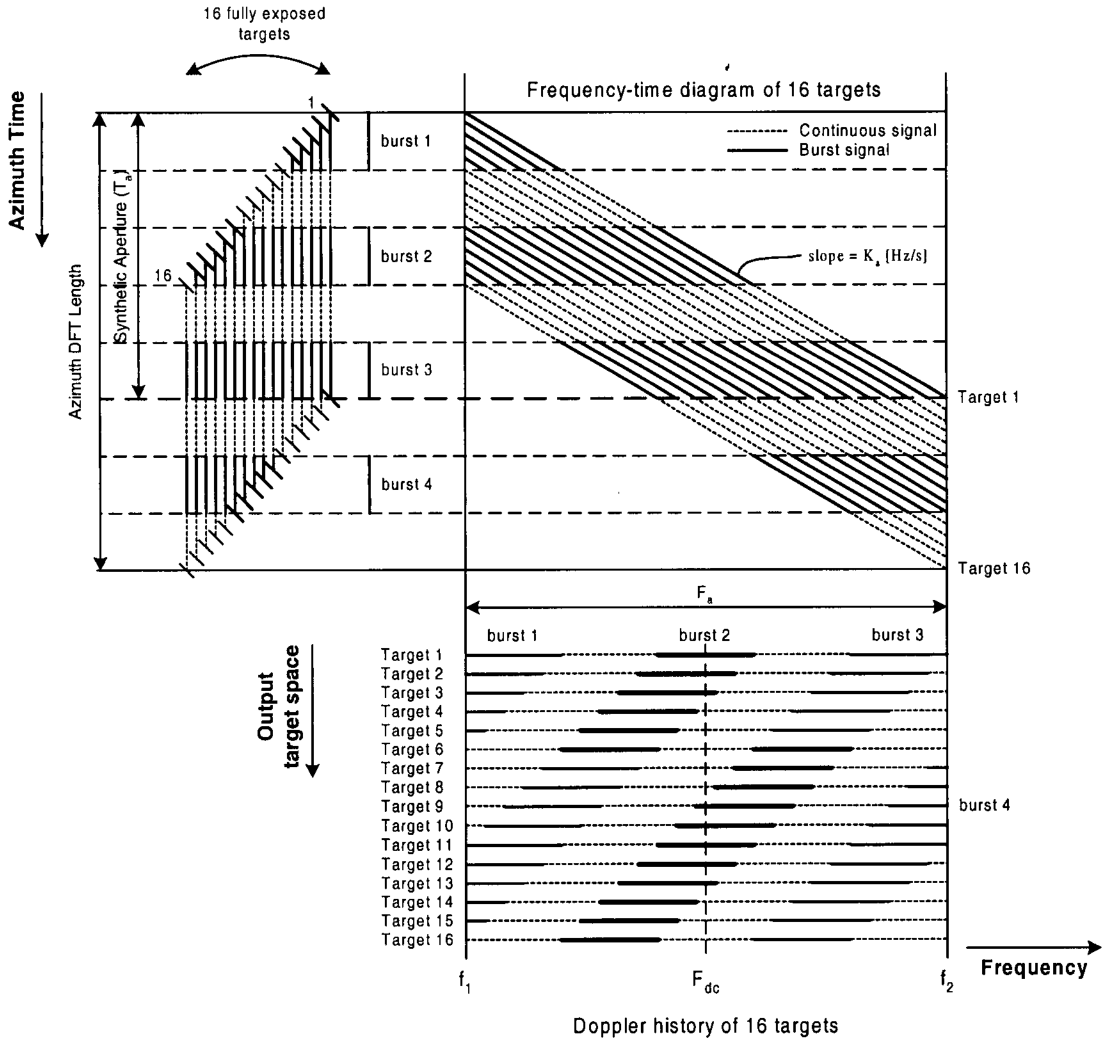


Figure 22 Burst-mode processing of 16 fully exposed and evenly spaced targets in one range cell

5.3 Properties of fully exposed targets in burst-mode processing

A typical 2-beam burst-mode data collection pattern is shown in Figure 22. Data from 16 evenly spaced, fully exposed targets in one range cell are shown, where the burst length is 20% of the aperture length. Dashed lines show the azimuth exposure time of each target if the SAR were operating in continuous mode, and the solid parts of each line show that part of each target actually exposed in burst mode.

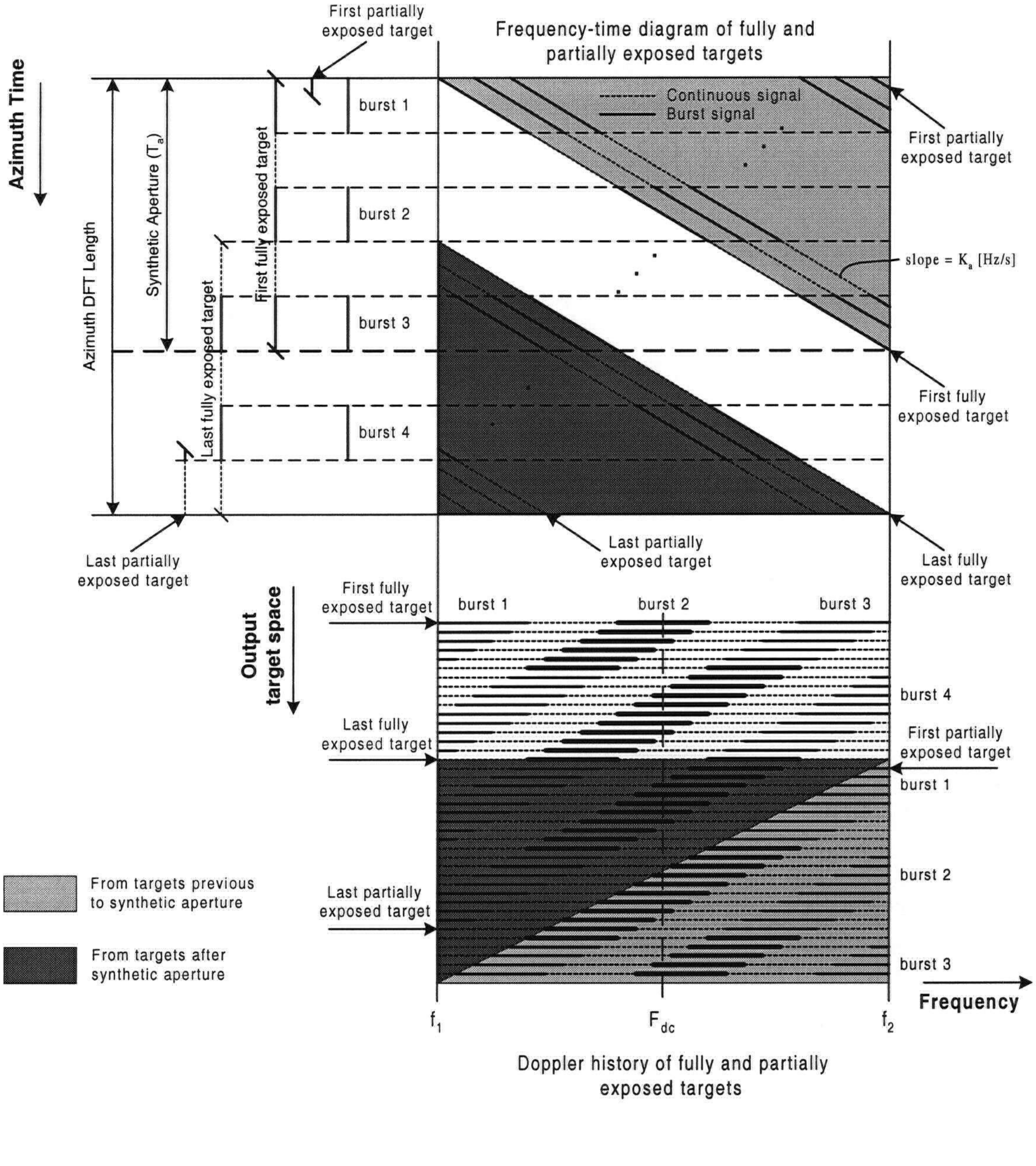
Note that the part of the target's exposure captured in burst mode varies with each target, which is illustrated in the frequency-time diagram of Figure 22. Each successive target is received at a lower Doppler frequency within a given burst, but is later captured at a higher frequency in the next burst, as long as it stays within the beam. The pulse repetition frequency (*PRF or* F_a) and the aperture length in azimuth time (T_a) are connected through K_a,

$$F_a = T_a K_a \text{ [Hz]} \tag{47}$$

T_a in the time-domain consist of 5 burst lengths, so F_a in the frequency-domain also consists of 5 burst bandwidths (Figure 22).

The Doppler history of the 16 targets is also shown in Figure 22, where it is seen that it takes up to 2 bursts to cover all of them. The plot shows the distribution of target spectral energy that would appear if an azimuth DFT were taken over the full 4 bursts, thus the DFT length was 4 bursts plus 4 gaps long in this case. Note that some targets appear in 2 full bursts (e.g. Target 6), some appear in 3 full bursts (e.g. Target 11), while others appear in two full and one partial burst. In this case, the average number of target exposures or bursts per aperture is 2.5. The number of bursts/aperture in ScanSAR systems is typically between 1.5 and 3.

If single-look complex processing is to be done, then there is a choice of which bursts to use for each target. Normally, the target exposures closest to the Doppler centroid (F_{dc}) will be selected, as shown by the heavier lines in the lower part of Figure 22. However, other bursts may be chosen, when the data is processed for InSAR purposes.



Azimuth Time
First partially exposed target
Frequency-time diagram of fully and partially exposed targets
Continuous signal
Burst signal
burst 1
burst 2
burst 3
burst 4
Azimuth DFT Length
Synthetic Aperture (T_a)
Last fully exposed target
First fully exposed target
First partially exposed target
slope = K_a [Hz/s]
First fully exposed target
Last partially exposed target
Last partially exposed target
Last fully exposed target
Output target space
First fully exposed target
burst 1
burst 2
burst 3
burst 4
Last fully exposed target
First partially exposed target
burst 1
burst 2
burst 3
Last partially exposed target
From targets previous to synthetic aperture
From targets after synthetic aperture
f_1
F_{dc}
f_2
Frequency
Doppler history of fully and partially exposed targets

Figure 23 Burst-mode processing of fully and partially exposed targets in one range cell

5.4 Properties of partially exposed targets in burst-mode processing

In the previous section the properties of fully exposed targets of a synthetic aperture were introduced when they are processed in burst-mode. These are the valid targets of continuous-mode processing: they have complete frequency-time history and get compressed using the full Doppler aperture (F_a) matched-filter.

Beside the fully captured targets, there are also partially exposed targets both at the leading edge and trailing edge of the azimuth DFT. These targets are incomplete, so they are discarded during continuous-mode processing. The frequency-time diagram and Doppler history of the partially exposed targets are shown in Figure 23, where the light gray region corresponds to signals from targets that begin previous to the start of the DFT and the dark gray region corresponds to signals from targets which end after the end of the DFT. Note that most of the partially exposed targets are also completely captured by one or two bursts, so they can be fully compressed using one burst width of their spectrum. In this way more targets with lower resolution can be fully compressed in the same synthetic aperture as in the continuous-mode for a given DFT length.

In Figure 23, both partially exposed regions in the Doppler history have a triangle shape in frequency-time space and they complete each other to a rectangular shape. Originally, the position of the light gray region is before the region of the fully exposed targets, corresponding to the position of targets to the synthetic aperture in azimuth time domain. The change in the position of the light gray region is caused by the wrap-around properties of the DFT.

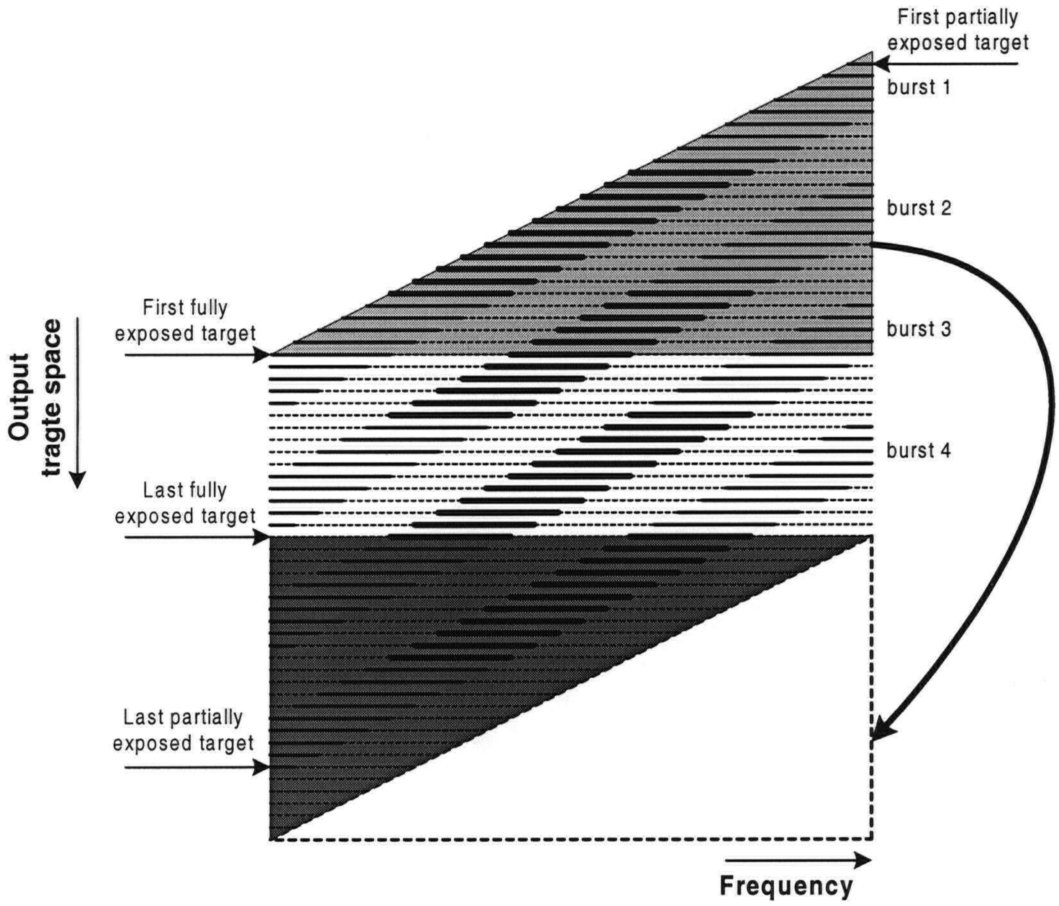


Figure 24 Effect of the circular convolution on the targets Doppler history

Figure 24 shows the original parallelogram region of the Doppler history and how the light gray region gets ‘wrapped-around’, when frequency-domain circular convolution is used instead of time-domain linear convolution. Note, targets in the parallelogram region get compressed into different output cells, while when the top triangle gets wrapped-around different targets can be compressed into the same cell depending on which Doppler sub-band is used. This property of the wrapped-around target Doppler history has to be taken under consideration, when selecting good targets during burst-mode signal compression using DFTs.

5.5 The SIFFT Algorithm

Most SAR processing algorithms are based on the fast convolution principle where a frequency-domain matched filter is used in the azimuth or Doppler frequency domain. When this method is applied to burst-mode data, the inter-burst gaps are filled with zeros and all the bursts are compressed at once using a full length matched filter followed by an IFFT. However, the compressed targets are then left with a burst-induced modulation.

The SIFFT algorithm differs from the conventional fast convolution algorithm in that short, overlapped IFFTs are taken after the matched filter multiply in the Doppler domain [22]. So, when one burst of a target is fully captured by the IFFT, little or no energy from adjacent bursts of the same target is present in the same IFFT. In this way, each IFFT compresses a group of targets without interference (modulation) from other bursts, and an accurate impulse response is obtained. The IFFT is acting like a band-pass filter to extract target energy from the segmented form of the targets' spectra. The filter is time varying in the sense that each successive IFFT is applied to a different frequency band.

To capture a target fully, the length of the IFFT must be at least as long as the length of the bandwidth of one burst. The bandwidth of one burst is

$$BW_{Hz} = \frac{N_b K_a}{F_a} \quad \text{[Hz]}$$

$$BW_{bin} = \frac{N_b K_a N_{FFT}}{F_a^2} \quad \text{[frequency bins]} \tag{48}$$

where N_b is the burst length and N_{FFT} is the length of the azimuth FFT in time samples. Then, the minimum IFFT length is

$$N_{IFFT\,\mathrm{min}} = BW_{bin} = \frac{N_b K_a N_{FFT}}{F_a^2} \text{ [samples]} \tag{49}$$

The IFFT cannot be longer than the bandwidth of one burst plus one gap, so that a fully-exposed target is not contaminated by a partial exposure of the same target at a different frequency. In the 2-beam ScanSAR case, the length of the gap is often equal to the burst length, so the maximum length IFFT is

$$N_{IFFT\,\mathrm{max}} = 2BW_{bin} = 2\frac{N_b K_a N_{FFT}}{F_a^2} \tag{50}$$

In practice, these length limits must be modified a little because of the spreading of target energy in the frequency domain, *i.e.* a guard band is used when locating the IFFTs. Note, $N_{IFFT\ max}$ and $N\ _{IFFT\ min}$ are proportional to K_a and N_{FFT}. The effect of this property is discussed in Section 5.6, where the arithmetic of the SIFFT is given. Usually, $N_{IFFT\ max}$ is smaller than N_{FFT}, so less than the whole Doppler spectrum is used for azimuth compression, which means that the output resolution of the SIFFT algorithm is smaller than the maximum available by a factor of N_{IFFT}/N_{FFT}.

Locations of $N_{IFFT\ max}$ and $N\ _{IFFT\ min}$ to extract targets with the highest energy using the closest burst spectrum to F_{dc} (thick lines) are shown in Figure 25. Targets, which are after or before the synthetic aperture and partially exposed, are noted as A1-A24 and B1-B24, respectively.

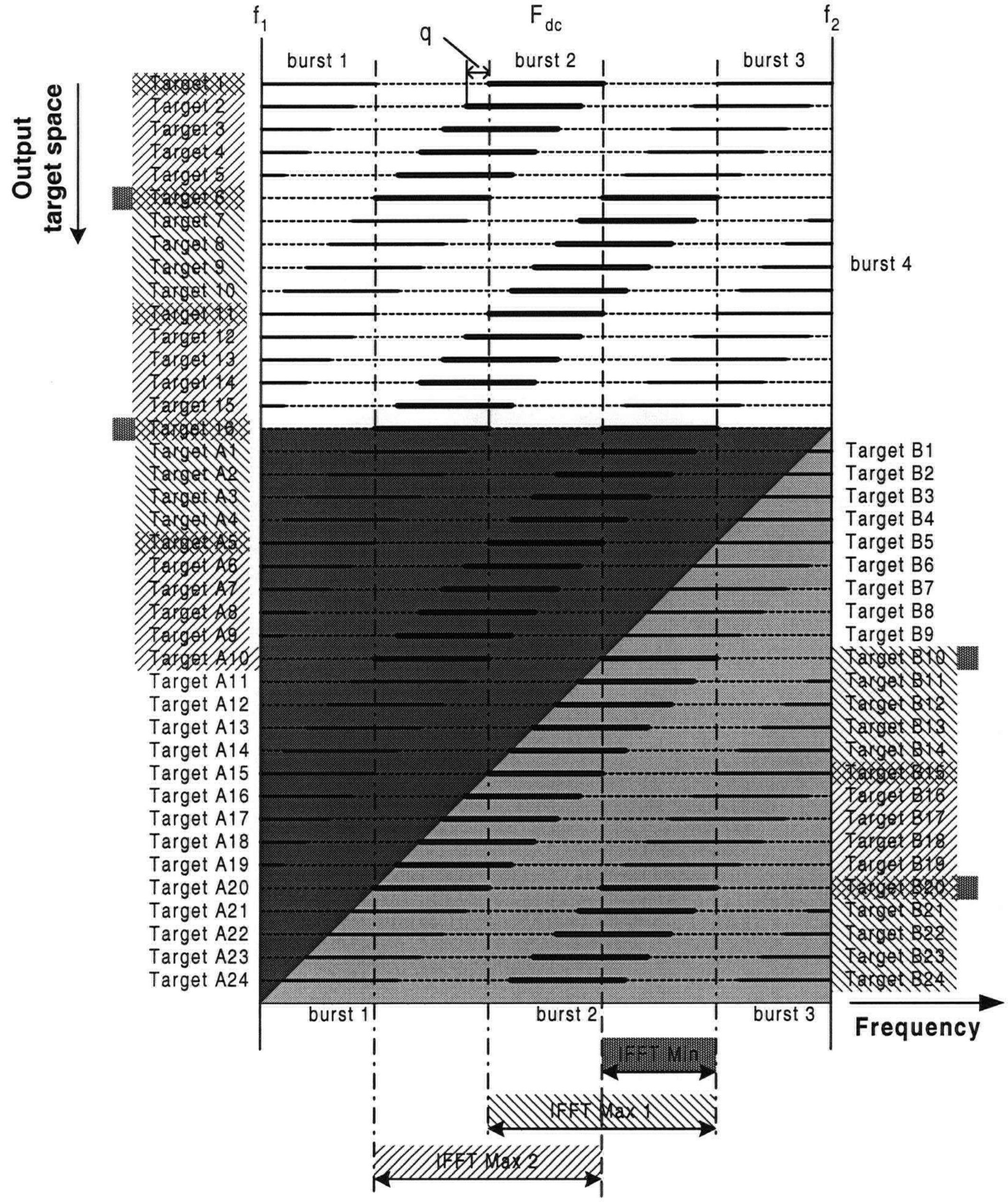


Figure 25 How minimum and maximum IDFT is placed to compress groups of targets from each burst

It can be seen that *IFFT Max 1* captures the complete energy of a single burst spectra of Targets 1, 6-11, 16-A5 B10-B15 and B20-24. For these targets, *IFFT Max 1* does not extract any energy from other bursts' spectra, so their impulse response is not corrupted by modulation. Similarly, *IFFT Max 2* captures the complete energy of a single burst spectra of Targets 1-6, 11-16, A5-A10 and B15-20 and between the two IFFTs, all the targets are correctly compressed. In order to form a continuous output image, the results of successive IFFTs are stitched together. If only bursts with the highest energy are used to compress targets, then each output target gets to a different output cell. Note, not all the partially exposed targets can be compressed with the best SNR and Targets A16-24 cannot be compressed at all, because none of the bursts covers them completely. Also note, targets before the full synthetic aperture (B10-B24) get compressed at last and their position in the output array is unique and correct.

5.5.1 Number of good output targets of a single IFFT

It was shown in the previous section that each IFFT compress different groups of targets correctly. We refer to these correct results as "good" output targets of one IFFT (G_{IFFT}). The General form of G_{IFFT} will be derived as below.

The number of groups of good output targets of an IFFT in the 2-beam burst-mode processing case is

$$N_{group} = \frac{N_{FFT}}{2N_b} \tag{51}$$

Note, N_{group} is equal to the number of complete bursts in a DFT. In our example in Figure 25, the DFT consists of 4 burst and 4 gaps, so there are 4 groups of good output targets.

Also note, that Target 1 and Targets B20-B24 are in the same group or burst (but Target B20-B24 is wrapped-around by the circular convolution).

The shift q between two consecutive targets in the Doppler frequency domain is shown on the top in Figure 25 and

$$q \;=\; \frac{K_a N_{FFT}^2}{F_a^2 N_{IFFT}} \text{ [frequency bins]} \tag{52}$$

Note, q is proportional to K_a and $N^{2}{}_{FFT}$, so the shift varies with range and the length of the DFT. Note, also that although targets are more than one samples apart in the Doppler domain, they placed one cell apart in the output space.

The number of good output targets in a group depends on the length of the IFFT and shift between targets:

$$G_{group} \;=\; \frac{N_{IFFT} - BW_{bin}}{q} \;+\; 1 \tag{53}$$

Note, when $N_{IFFT} = N_{IFFT\,MIN}$, then $G_{group} = 1$ as it is shown in Figure 25.

Then, the number of good output targets from an IFFT can be expressed as:

$$\begin{aligned} G_{IFFT} &= G_{group} \cdot N_{group} \\ &= \frac{N_{FFT}}{2N_b}\left(\frac{(N_{IFFT} - BW_{bin})F_a^2\, N_{IFFT}}{K_a\, N_{FFT}} + 1 \right) \end{aligned} \tag{54}$$

5.5.2 Real data simulation of burst-mode processing

A real data experiment was done to verify the above described properties of the SIFFT algorithm. A burst-mode processor was created by combining the MDA/UBC dtSAR processor and our Matlab azimuth processing program.

In the experiment, the raw data ingestion, range compression and RCMC are done by the dtSAR processor. The RCMC-ed data, along with the required processing parameters, such as F_a, K_a and the Doppler centroid frequency are read into our Matlab program for azimuth processing. Before we can apply the burst-mode algorithm to the dtSAR output data, we have to generate the burst mode signal and correct the antenna pattern to avoid scalloping in the output. The antenna pattern correction is done by summing the azimuth FFT of each range cell, and polyfitting the summed data.

The burst mode data is emulated from SAR continuous-mode data by windowing the signals in the azimuth direction in the Matlab program. The parameters of the ERS-1 data are shown in Table 6. The single look detected SAR image generated with the SIFFT algorithm is shown in Figure 26.

Processing parameter	Value	Unit
Sampling frequency (F_a)	1679.90	[Hz]
Doppler Centroid (F_{dc})	447.01	[Hz]
Burst length (N_b)	274	Samples
Range cells	1024	Samples
Azimuth samples (N_{FFT})	2048	Samples

Table 6 ERS-1 parameters for real data simulation

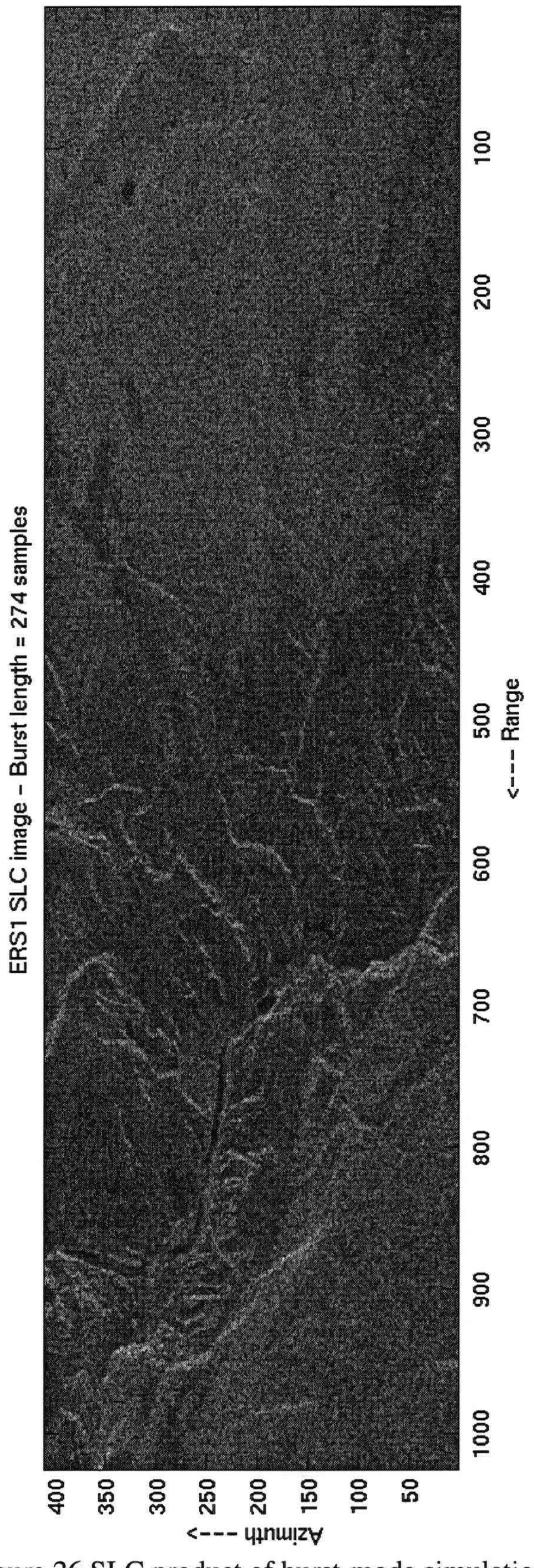


Figure 26 SLC product of burst-mode simulation

Although, the synthetic aperture length varies with range, the burst length is approximately 1/5th of the aperture through the whole processing region. During the azimuth compression, $N_{IFFT\ MIN}$ was used for bursts close to the Doppler centroid, so the burst-mode data is extracted with the best SNR. The resolution of the output image is 1/5th of the original continuous SLC, because maximum only 1/5th of the targets' Doppler spectrum can be used for signal compression.

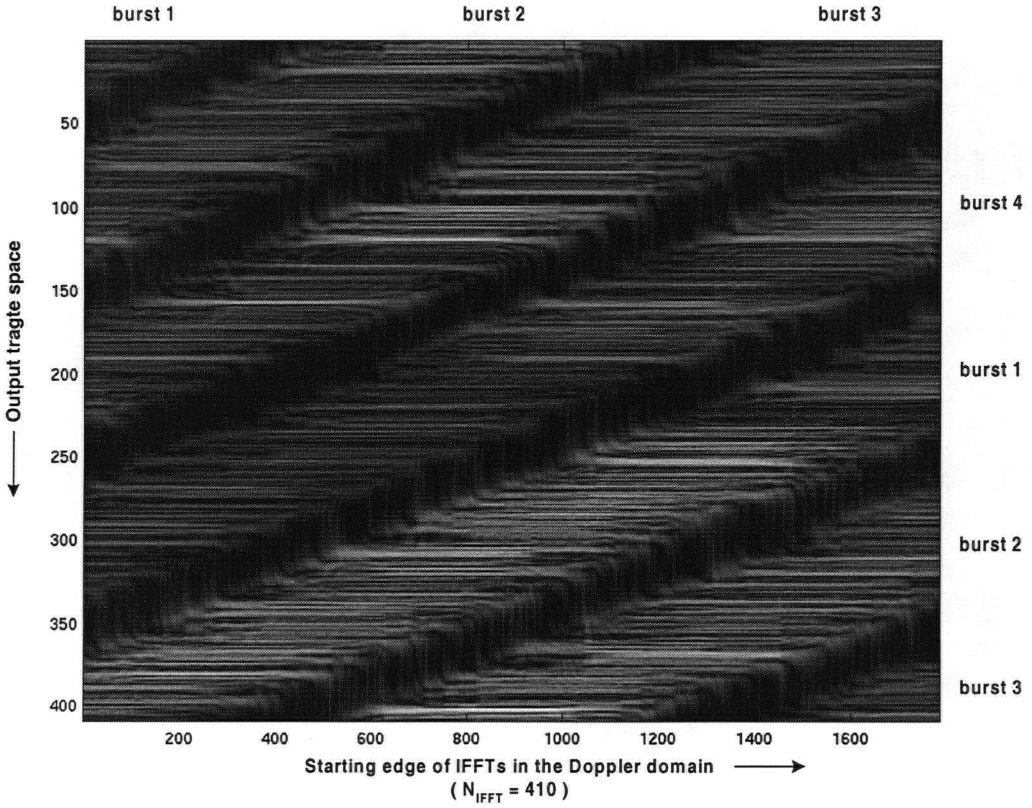


Figure 27 The Doppler history of real burst-mode data

The Doppler history of one range cell of the real data was generated using the IMFT algorithm is shown in Figure 27. IMFTs, with size of $N_{IFFT\ MIN}$ were taken, at each frequency bin, and the result of the transforms were stored in a matrix. Figure 27 shows the values of the matrix, where the brightness of the image corresponds to the magnitude of the output result. The bright points represents the good output answers, while the darker parts show the gaps in the frequency domain. Note, the energy leakage of the good output points in the figure. When the IMFT is applied to process a data record, it needs *N* samples to fully contain data. So, in the output space there is an *N* sample long ramp before the first valid result. Similarly, after the last valid result there is an N samples long ramp as the IMFT is sliding off from the data. These properties of the frequency analysis cause the leakage in the output space. Ideally Figure 27 would look like the Doppler history in Figure 23, thus it would have the same intervals of gaps and targets.

5.6 Efficiency of SIFFT using the IMFT and the IFFT algorithms

To show the efficiency of the IMFT vs. the IFFT algorithm we use the parameters of the alternating polarization (AP) mode of ESA Envisat satellite. Alternating polarization mode provides high resolution products (approximately 30m) in any of the seven swath located over a range of incidence angles spanning from 15 to 45 degrees with polarization changing from sub-aperture to sub-aperture within the synthetic aperture. Effectively, a 2-beam case ScanSAR technique is used but without varying the sub-swath.

The velocity of the satellite V_r = *6700 m/s*, the radar wavelength λ = *0.0567 m* and the azimuth FFT is 2048 and 4096 samples long during the efficiency evaluation. Other parameters of the seven swath are given in Table 7.

Swathes	Sampling frequency F_a [Hz]	Burst/Gap length in [samples]	Range [km]
IS 1	1678	194	825 – 864
IS 2	1645	196	843 – 891
IS 3	2096	257	887 – 934
IS 4	1680	218	929 – 990
IS 5	2082	277	983 – 1032
IS 6	1698	238	1027 – 1087
IS 7	2070	297	1080 – 1133

Table 7 Envisat swath parameters

5.6.1 Effect of varying SAR parameters and SNR/efficiency tradeoffs

As it was shown previously, the azimuth FM rate of the received SAR signal is inversely proportional to range, so it changes as the range varies in each range cell. While the length of the bursts is constant in the azimuth time domain, the bandwidth of the bursts (49) varies with range because it is proportional to K_a. Table 8 shows the maximum and minimum value of the burst bandwidth of each swath in Hz and in frequency bins. At close range the bandwidth has its maximum, while at far range has its minimum as it is shown in Figure 28 for swath IS1. In a swath, the minimum IFFT length ($N_{\text{IFFT min}}$) should be at least as long as the maximum BW_{bin} to compress all the targets in each range cell correctly. Note, the lower and upper limit length of the IFFTs related to the burst bandwidth so they are different for each swath as well.

Swathes	BW_{Hz} [Hz]		BW_{bin} [frequency bins]			
			N_{FFT} = 2048		N_{FFT} = 4096	
	MIN	MAX	MIN	MAX	MIN	MAX
IS 1	212	222	259	271	517	542
IS 2	212	224	264	279	527	557
IS 3	208	219	203	214	406	428
IS 4	208	221	253	270	506	539
IS 5	204	214	201	211	402	422
IS 6	204	216	246	261	493	521
IS 7	201	210	198	208	397	416

Table 8 Minimum and maximum burst bandwidth of the seven swathes

The SNR of the SIFFT algorithm depends on the ratio of the IFFT and the burst bandwidth [22]. The SNR is maximum when the IFFT length is equal to the burst bandwidth ($N_{IFFT} = N_{IFFT\,min}$), while the SNR is 3 dB lower when the IFFT window is one burst plus one equal-sized gap long ($N_{IFFT} = N_{IFFT\,max}$). So, in order to keep the SNR loss below a certain value across the swath, there is a need to choose specific azimuth IFFT lengths. The change in SNR for a given IFFT length is:

$$dSNR = 10\log_{10}\left(\frac{N_{IFFT}}{BW_{bin}}\right) \tag{55}$$

If we choose N_{IFFT} as small as possible at near range and have it stay the same through the whole swath, the SNR will decrease, as the burst bandwidth decreases with range.

Figure 28 shows how *dSNR* changes with range, in the case of IS1 swath and N_{FFT} = *2048*. The decrease in SNR is zero at near range and rises to about 0.2 dB at far range. Note, although BW_{bin} in (54) depends on N_{FFT}, the gradient and maximum of *dSNR* is the same for different azimuth FFT lengths.

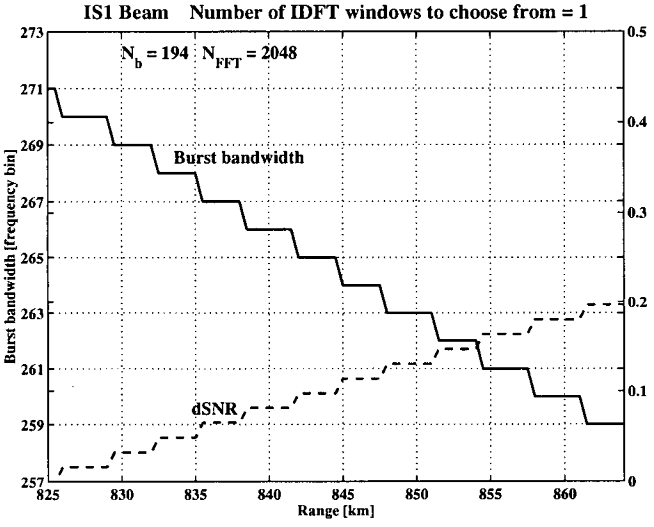


Figure 28 Burst bandwidth and *dSNR* of IS1 swath

5.6.2 Arithmetic of the SIFFT algorithm using the IMFT and the IFFT algorithms

During the efficiency evaluation of the SIFFT algorithm, the full-IMFT, the reduced-IMFT and the mixed-radix IFFT algorithms are considered. A formula of the arithmetic of each algorithm is introduced below.

It can be shown that the number of IFFTs to compress bursts with the highest energy is

$$NIFFT = 2\frac{N_{IFFT}}{N_{FFT}}\,N_b\,\frac{1}{G_{group}}$$
$$= \frac{2N_{IFFT}\,N_b\,q}{N_{FFT}\,(N_{IFFT} - BW_{bin} + q)}$$

(56)

We can see from (56) that *NIFFT* inversely proportional to G_{group}, thus if more targets are in a group less IFFTs are needed for data extraction. Than, the number of operations needed to compress all the targets using the IFFT algorithm is

$$OPIFFT = NIFFT \cdot OPN_{IFFT}$$

(57)

where OPN_{IFFT} is the number of operations needed for one *N* sample long mixed-radix IFFT. Note if *N* is power of 2 than $OPN_{IFFT} = 5\,N\,log_2(N)$.

As we saw in section 1.4.2, in case of the full-IMFT algorithm $OPIMFT = M(8N_{IFFT} + 2)$ real operations are needed to analyze an M-point complex data record. In the case of 2-beam burst processing $M = 3BW_{bin}$, so the arithmetic of the full-IMFT algorithm is

$$OPIMFT_{Full} = 3\,BW_{bin}\,(8\,N_{IMFT} + 2)$$

(58)

During the IDFT extraction in the SIFFT algorithm, only a portion of the output target space – 'good' output targets (G_{IFFT}) - is compressed correctly. Although the amount of the 'good' targets remains the same through the processing of a range cell, but their position changes with the position of the IDFTs. So, a simple reduced-IMFT algorithm cannot be used for the target extraction. The position of the computed Doppler - spectrum coefficients has to change in phase with position of the not-compressed good output samples, and the Doppler-frequency coefficients of the already compressed targets do not have to be computed during the rest of the processing. The arithmetic of the required reduced-MFT algorithm is given below.

At first, the reduced - IMFT algorithm has to be applied N_{IMFT} times to give the first valid compression result. This requires

$$OPIMFT_{Reduced\,1} = N_{IMFT}(8\,N_{IMFT} + 2) \tag{59}$$

real operation. When the first IMFT is done, G_{IFFT} number of targets are compressed correctly, so in the next IMFT $N_{Reduced\ 1} = N_{IMFT} - G_{IFFT}$ number of targets have to be compressed. Than, the IMFT window is shifted q times, a sample at a time, till it fully covers the next target in the group. Now, G_{group} number of targets can be extracted correctly, so during the next q shift $N_{Reduced}$ - G_{group} targets need to be extracted. The deduction of the spectrum coefficients from the previously reduced whole output target cell repeats $(3BW_{bin} - N_{IFFT})/q$ times trough the processing region, until all the targets get compressed. It can be shown that the arithmetic of this procedure is:

$$OPIMFT_{Reduced\,2} = 2[3BW_{bin} - N_{IMFT}][4(N_{IMFT} - G_{IFFT}) - 2G_{IFFT}(\frac{3BW_{bin} - N_{IMFT}}{q} - 1) + 1] \tag{60}$$

Using the previously introduced equations

$$\begin{aligned} OPIMFT_{Reduced} &= OPIMFT_{Reduced\,1} + OPIMFT_{Reduced\,2} \\ &= N_{IMFT}(8N_{IMFT}+2) + \\ &\quad 2[3BW_{bin} - N_{IMFT}][4(N_{IMFT} - G_{IFFT}) - 2G_{IFFT}(\frac{3BW_{bin} - N_{IMFT}}{q} - 1) + 1] \end{aligned} \tag{61}$$

real operations are needed to compress all targets using the reduced-IMFT algorithm. Note that the arithmetic of the reduced-IMFT depends on G_{IFFT} and G_{group}, thus if more targets get extracted by an IMFT than less computation is needed. Note, the implementation of the reduced-IMFT is the same as the full-IMFT algorithm, except the timing of the position of the output targets need to be compressed.

During the efficiency evaluation, we choose the IDFT lengths on the principles that we want:

- maximum SNR at near range,
- minimum sampling rate at near range, and
- the sampling rate constant with range,

and we set the forward azimuth FFT to be 2048 and 4096 samples long.

First, we make the IDFT as small as possible at near range (*i.e.* $N_{IDFT} = Max\ BW_{bin}$), and have it stay the same with range, even though the burst bandwidth decreases with increasing range. Thus there is only one IDFT window length to choose from ($w = 1$) for the IFFT or the IMFT algorithms.

Second, we consider the case where the IDFT is allowed to be up to 4 samples longer than the minimum (*i.e.* $Max\ BW_{bin} \leq N_{IDFT} < Max\ BW_{bin} + 4$). This allows some flexibility in choosing a favorable IFFT length from 5 different window sizes, at the expense of a small decrease in SNR. Table 9 shows the IDFT lengths and the maximum of the corresponding *dSNR* for the seven swathes of the Envisat.

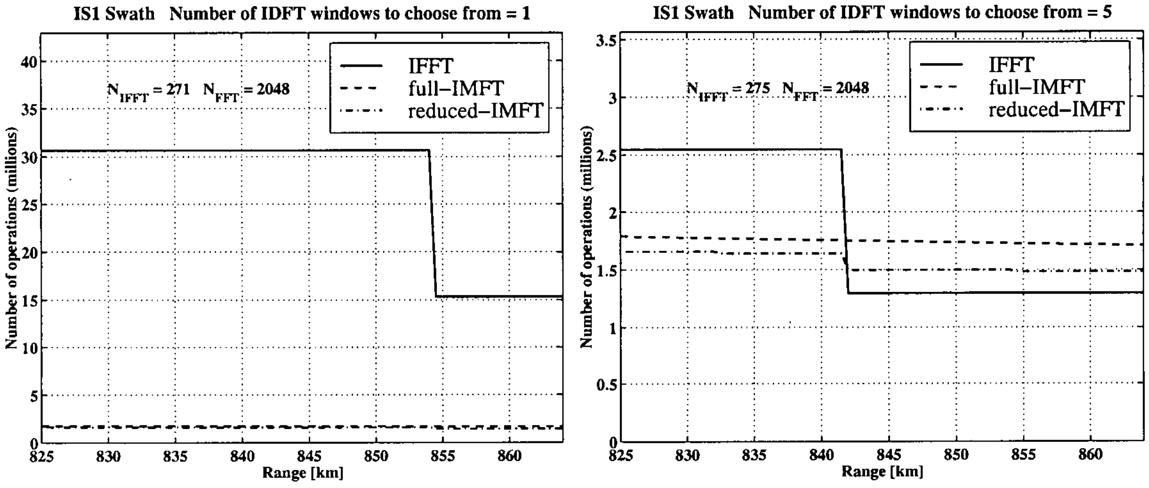
Swathes	$N_{FFT} = 2048$				$N_{FFT} = 4096$			
	w = 1		w = 5		w = 1		w = 5	
	N_{IDFT}	Max dSNR	N_{IDFT}	Max dSNR	N_{IDFT}	Max dSNR	N_{IDFT}	Max dSNR
IS 1	271	0.20	275	0.26	542	0.20	546	0.24
IS 2	279	0.24	280	0.26	557	0.24	560	0.26
IS 3	214	0.23	216	0.27	428	0.23	432	0.27
IS 4	270	0.28	270	0.28	539	0.27	540	0.28
IS 5	211	0.21	215	0.29	422	0.21	425	0.24
IS 6	261	0.26	264	0.30	521	0.24	525	0.27
IS 7	208	0.21	210	0.26	416	0.20	420	0.24

Table 9 The length of the IDFTs and the corresponding dSNR

The burst bandwidth is direct proportional to N_{FFT} (48), so it changes with the same ratio as N_{FFT} does. Note that if the value of BW_{bin} is bigger, than it is easier to find a highly composite number in its neighborhood, thus it is easier to pick a suitable window length for the IFFT algorithm. Also note in Table 9, that the change in *dSNR* when a more suitable window is used for the IDFTs is less than 0.1 dB, thus the SNR decrease is practically negligible.

Figure 29 shows the number of real operations of the SIFFT algorithm when it is used to compress the data of IS1 swath. The IDFT algorithms used to obtain the azimuth compression are the mixed-radix IFFT and the full- and reduced-IMFT. In Figure 29 (a), we see that the IMFT algorithms are more efficient trough the whole swath when only the minimum window length can be used.

The IFFT window begins to cover the bandwidth of more than one target as BW_{bin} decreasing with range, so there is a change in the value of G_{group} and G_{IFFT} when the IFFT window starts to fully cover two or more targets in the Doppler domain. The IMFT arithmetic decreasing slowly with rage and the arithmetic of the mixed-radix IFFT drops down to its half when G_{group} doubles from 1 to 2. The IFFT arithmetic is constant on both side of the down-step. Note that there is more than a factor of 10 difference between the arithmetic of IMFT and IFFT, because the window length is prime (271), so only the direct IDFT can be used to obtain the IFFT results.



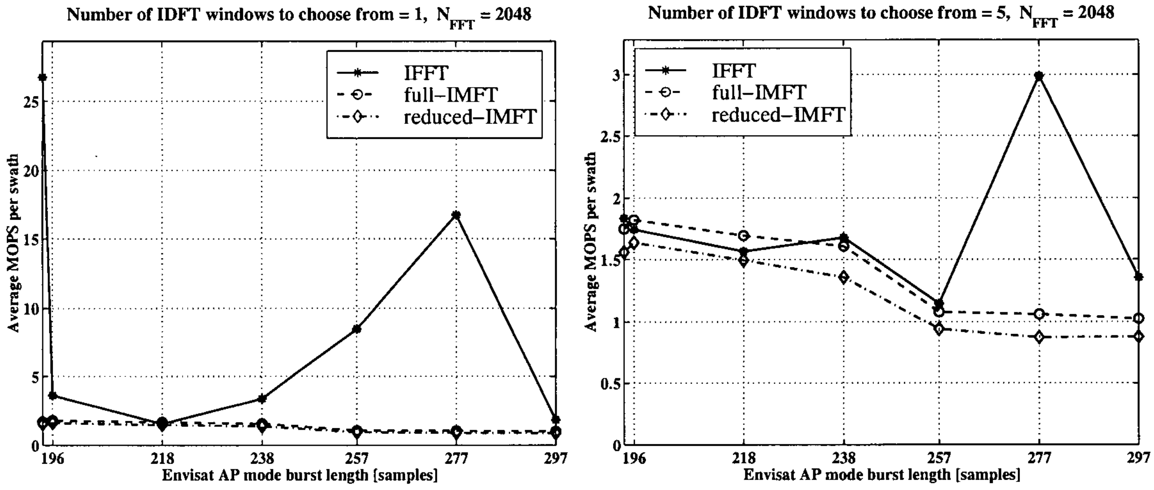
a)

b)

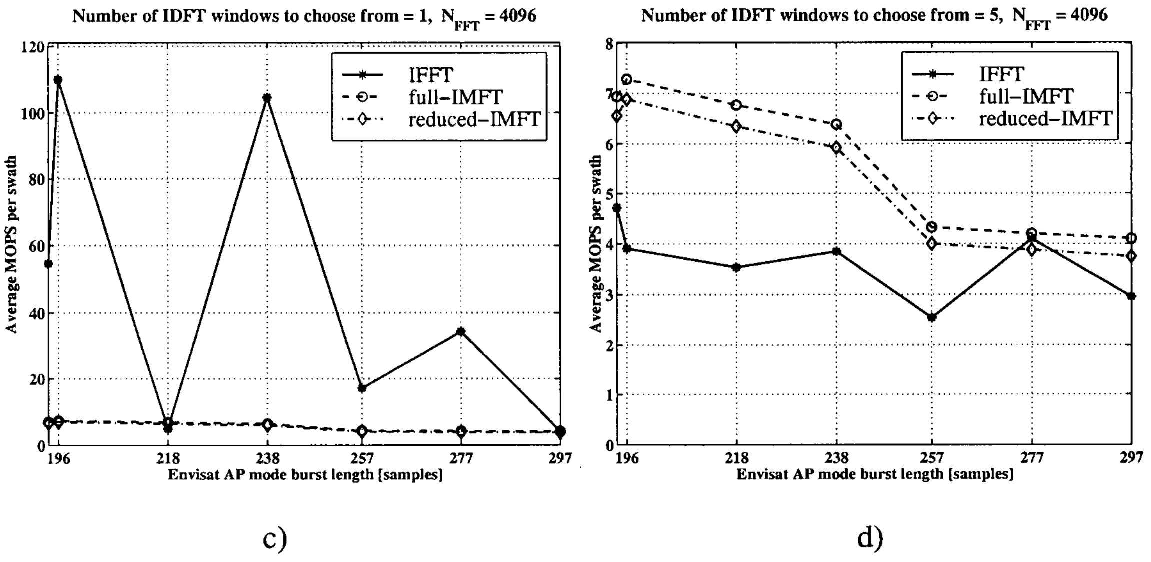
Figure 29 Arithmetic of the SIFFT algorithm when applied to the IS1 swath

In Figure 29 (b), we can see that the mixed-radix IFFT arithmetic dramatically drops down when the IFFT window length can be a composite number (275). The arithmetic of the IMFT algorithms did not change significantly and they are efficient only in a part of the swath, where only one target can be fully compressed in each group. The down-step of the IFFT arithmetic happens in closer range, because the IFFT window is larger, so it starts to fully cover two targets earlier. Note that the reduced-IMFT arithmetic also drops down when G_{group} doubles, but this change is not significant compare to the change in the IFFT arithmetic. Note, the arithmetic of the IMFT and IFFT algorithms of the other swaths is some what similar to the arithmetic of swath IS1.

In figure 30, the average millions of operations (MOPS) are shown for all the Envisat swathes when the azimuth FFT 2048 and 4096 samples long. The trend of the arithmetic of the full- and reduced-IMFT is similar in all cases, while the IFFT arithmetic is quite variable, depending upon the composition of the window length. When $N_{FFT} = 2048$, the IMFT is more efficient for most of the swathes even if there is an option to chose a suitable window length for the IFFT algorithm. When $N_{FFT} = 4096$, the IMFT is almost always better than the IFFT if only the smallest IDFT window can be used (Figure 30 (c)). When there is a possibility to chose a favorable IFFT length, the IFFT is more efficient for all the swathes except one. Note, there is a higher possibility to find a high composite number in the neighborhood of the smallest window size value and more group of good targets (51) can be extracted, when the azimuth FFT larger. So, the arithmetic of the IFFT algorithm decreases more dramatically, than in the case of smaller N_{FFT} length.



a) b)



c)

d)

Figure 30 Arithmetic of the SIFFT when it is applied to Envisat AP burst mode operation

From the above given arithmetic survey we can see that the IMFT algorithm can improve the computational efficiency of the SIFFT algorithm when,

- the azimuth FFT is relatively small,
- the maximum near range SNR is required and
- the IDFT window length is a non-composite number.

Beside its efficiency, the IMFT has the following advantages when it is applied to the SIFFT algorithm:

- the IMFT has more consistent computing load as the burst bandwidth changes with range and
- it is easier to implement the IMFT algorithm for different burst and N_{FFT} lengths, because the same IMFT algorithm can be used for the different IDFT window lengths.

Chapter 6

Conclusions

6.1 Summary

The objective of this work has been to further develop the theory of the momentary Fourier transform, to examine its arithmetic and efficiency and to show what advantages it offers when it is applied to the SPECAN SAR algorithm and the SIFFT burst-mode data processing algorithm.

The momentary matrix transform was introduced and it was shown when it takes the form of the DFT or the IDFT, the resulting MFT/IMFT have an efficient recursive computational structure. The spectrum coefficients of the MFT/IMFT can be calculated independently and only one complex multiplication and two complex additions are needed to update each spectrum component. This is a factor of log_2N improvement over the radix-2 FFT algorithm if all incremental DFT results are needed. The efficiency of the MFT/IMFT do not rely upon the transform length being a power of two, in contrast to standard FFT algorithms. The MFT/IMFT transformation pair can provide more efficient computation of the DFT when:

- DFTs are highly overlapped
- only a few Fourier coefficients are needed
- a specific, non-composite DFT length is needed,

and they can be useful in different applications of signal processing such as:

- on-line computations in real-time spectral analysis
- on-line signal identification and detection
- speech processing
- radar and sonar processing
- narrow-band filtering.

After the introduction of the SPECAN SAR processing algorithm, the applicability of the MFT to SPECAN has been investigated. Although, the MFT does not improve the computational efficiency of the SPECAN algorithm, it has the following advantages over the radix-2 and mixed-radix FFT when they are applied to SPECAN:

- The MFT has consistent computing load as the DFT length changes.
- It is easier to implement the MFT algorithm for variable window length. The architecture of a SPECAN processor using MFT is less complex, because the same MFT algorithm can be used for the different window lengths.
- The full radar resolution can be achieved, because the full Doppler spectrum of the targets can be used for the compression by using high-overlapped DFTs.

- The output sampling rate of SPECAN is more uniform.

In burst-mode SAR processing, the time-varying spectral properties of the azimuth received data requires that highly-overlapped inverse DFTs used at specific points in the spectral domain to obtain the azimuth compression. It was shown that the IMFT can be more efficient than the IFFT when it is applied to the SIFFT burst-mode data processing algorithm, when:

- the azimuth FFT is relatively small,
- the maximum near range SNR is required and
- the IDFT window length is a non-composite number.

6.2 Future work

The research of this thesis project raised the following topics for further study:

- Further develop the theory of the momentary matrix transform, and see what other discrete transforms, which are used in signal processing can be efficiently implemented using the MMT.
- Investigate if there is a closed recursive formula of the MMT for shifts greater than one, using higher order permutational matrixes.
- Further examine the applicability of MFT to other SAR processing techniques, such as interferometry SAR (InSAR) processing. The MFT can improve the computational efficiency of the local frequency estimation of interferograms, because frequency estimates are needed at every sample point of the interferogram. In addition, there is a possibility of obtaining better resolution estimates of the local frequency using two

passes of MFT, in order to provide more accurate estimates of unwrapped interferogram phase.

- Compare the accuracy, signal-to-noise ratio and computational efficiency of the SPECAN and the SIFFT algorithm, when they are applied to burst-mode data processing.
- Investigate how efficiently and with what parameters the SIFFT algorithm could be used to process continuous-mode SAR data.

Bibliography

[1] A. Papoulis, *Signal Analysis*. McGraw-Hill, 1977

[2] G. Strang, *Linear Algebra and Its Applications*. Saunders College Publishing, Third Edition, 1988

[3] J. Curlander and R. McDonough, *Synthetic Aperture Radar: System and Signal Processing. Wiley*, New York, 1991.

[4] J. G. Proakis and D.G. Manolakis, *Digital Signal Processing*. Prentice Hall, Third Edition, 1996

[5] R. R. Bitmead and B. D. O. Anderson, "Adaptive frequency sampling filters," *IEEE Trans. On Circuits and Systems*, vol. CAS-28, pp. 524-534, June 1981.

[6] J. Dudás, *The Momentary Fourier Transform.* Ph.D. thesis, Technical University of Budapest, 1986.

[7] H. Lilly, "Efficient DFT-based model reduction for continuous systems", *IEEE Trans. on Automatic Control*, vol.36, pp. 1188-1193, Oct. 1991.

[8] B. G. Sherlock and D. M. Monro, "Moving discrete Fourier transform", *IEE Proceedings-F*, vol. 139, pp. 279-282, Aug. 1992.

[9] S. Albrecht, I. Cumming and J. Dudás, "The momentary Fourier transformation derived from recursive matrix transformations", in *Proceedings of the 13th International Conference on Digital Signal Processing, DSP'97*, (Santorini, Greece) vol. 1, pp. 337-340, July, 1997.

[10] I. Cumming and J. R. Bennett, "Digital processing of SEASAT SAR data", *IEEE 1979 International Conference on Acoustic, Speech and Signal Processing*, (Washington, D.C., USA), April 2-4, 1979.

[11] I. Cumming and J. Lim, "The design of a digital breadboard processor for ESA remote sensing satellite synthetic aperture radar", *technical report, MacDonald Dettwiler, Richmond, BC, Canada,* July 1981. Final report for ESA Contract No. 3998/79/NL/HP(SC).

[12] M. Sack, M. Ito, I. Cumming, "Application of Efficient Linear FM Matched Filtering Algorithms to SAR Processing", *IEEE Proc-F*, vol.132, no. 1, pp. 45-57, 1985.

[13] T. Ngo and C. M. Vigneron, "Project Report: UBC SQLP v.1.7.", *technical report* Radar Remote Sensing Group, Electrical and Computer Engineering, University of British Columbia, 1995.

[14] H. Hobooti, "Radiometric Correction in Range-SPECAN SAR Processing", *M.A.Sc. thesis*, Electrical and Computer Engineering, University of British Columbia, April 1995.

[15] Sandor Albrecht, "The Momentary Fourier Transform and Its Application to the SPECAN SAR Processing Algorithm", *technical report SA-97-01*, Radar Remote Sensing Group, Electrical and Computer Engineering, University of British Columbia, 1997.

[16] S. Albrecht and I. Cumming, " Application of Momentary Fourier Transform to the SPECAN SAR Processing Algorithm", in *Proceedings of the IX. European Signal Processing Conference, EUSIPCO'98*, (Rhodes, Greece) September 8-11, 1998.

[17] F. Wong, "Processing Envisat AP mode with Range Doppler algorithm", *technical report, MacDonald Dettwiler, Richmond, BC, Canada,* January 1996.

[18] I. Cumming Y. Guo and F. Wong, "Analysis and Precision Processing of Radarsat ScanSAR Data", *In Geomatics in the Era of Radarsat, GER'97*, (Ottawa, Canada), May 25-30, 1997.

[19] F. Wong, D. Stevens and I. Cumming, "Phase-Preserving Processing of ScanSAR Data with Modified Range Doppler Algorithm", in *Proceedings of the International*

Geoscience and Remote Sensing Symposium, IGARSS'97, (Singapure), pp.725-727, August 3-8, 1997.

[20] I. Cumming Y. Guo and F. Wong, "A Comparison of Phase-Preserving Algorithms for Burst-mode ScanSAR Data Processing", in *Proceedings of the International Geoscience and Remote Sensing Symposium, IGARSS'97*, (Singapure), pp.731-733, August 3-8, 1997.

[21] Y. Guo, "Precision Processing of Burst-Mode ScanSAR Data", *M.A.Sc. thesis*, Electrical and Computer Engineering, University of British Columbia, October 1997.

[22] I. Cumming Y. Guo and F. Wong, "Modifying the RD Algorithm for Burst-mode SAR Processing", in *Proceedings of the European Conference on Synthetic Aperture Radar, EUSAR'98*, (Friedrichshafen, Germany), pp.477-480, May 25-27, 1998.

[23] S. Albrecht and I. Cumming, " The Application of Momentary Fourier Transform to SIFFT SAR Processing ", in *Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis*, (Pittsburgh, USA) October 7-9, 1998.