Spring 2008

Optimal and Permissible Sampling Rates for First-Order Sampling of Two-Band Signals

Daniel Frederick Linder

OPTIMAL AND PERMISSIBLE SAMPLING RATES
FOR FIRST-ORDER SAMPLING OF TWO-BAND SIGNALS

by

DANIEL F. LINDER II
(Under the Direction of Yan Wu)

ABSTRACT

Sampling theory plays an essential role in the advancement of digital signal processing (DSP). All known DSP processors only work with digital samples of an analog signal (continuous-time signal). Therefore, reliable sampling of a signal is crucial for the successive phases of DSP. A well-known industry standard for sufficient sampling of an analog signal is that the sampling rate is at least twice the highest frequency of the signal. Obviously, the greater the highest frequency of the signal, the higher the sampling rate required, hence, more wear and tear on the sampling device. This research focuses on developing sampling methods for passband signals, which arises for broad-band signal processing, and it has drawn great interests in the DSP community. A first-order sampling method with optimal and total identification of all permissible sampling rates for two-band passband signals is studied in this work. A rigorous proof for all the sampling rates is presented. It is shown that the new sampling rates are much lower than the industrial standard. Therefore, the new sampling mechanism has sound theoretical and commercial values. Quantitative analysis is performed on the proposed sampling method, including a fast algorithm for computing all feasible sampling rates for two-band passband signals.

INDEX WORDS: Nyquist rate, Digital signal processing, Sampling, Passband, Signal

OPTIMAL AND PERMISSIBLE SAMPLING RATES FOR FIRST-ORDER SAMPLING OF TWO-BAND SIGNALS

by

DANIEL F. LINDER II

B.S., Georgia Southern University, 2006

A Thesis Submitted to the Graduate Faculty of Georgia Southern University in Partial

Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

STATESBORO, GEORGIA
2008

OPTIMAL AND PERMISSIBLE SAMPLING RATES FOR FIRST-ORDER SAMPLING OF TWO-BAND SIGNALS

by

Daniel F. Linder II

Major Professor: Yan Wu

Committee: Yan Wu
David Stone
Goran Lesaja

Electronic Version Approved:
May 2008

ACKNOWLEDGEMENTS

First, I would like to thank my mother, who has encouraged me to strive toward an admirable standard in my endeavors. Her input and advice have been extremely beneficial, and have given me insight into academic directions.

This work is a product of the friendship that I have come to enjoy over the past few years with Dr. Yan Wu. He is a very passionate teacher, who is student achievement oriented and enjoys involving and sharing projects with his undergraduate and graduate students. His careful planning and insight has helped me explore new areas of mathematics during this research project and a previous one two years earlier. My deepest thanks go to him and I am greatly advantaged to have had the chance to work under him.

I would also like to thank the members of the graduate committee, Dr. Goran Lesaja and Dr. David Stone, who have been crucial in the review and development of the work. I am privileged to have had both for numerous classes in the past and know they have made significant impacts on my thoughts of mathematics. My work has been greatly influenced by the three.

TABLE OF CONTENTS

LIST OF FIGURES

NOMENCLATURE

Δ	stepsize of shifting
$I_{[a,b]}^{m\Delta}$	closed interval for $[a+m\Delta, b+m\Delta]$
$[a,b]$, $[d,c]$	Two closed intervals with $d<c<a<b$
$\lfloor\ \rfloor$	the floor function
$\in$	Belongs to
$\exists$	There exists
Z^+	Positive integers
N	Natural numbers
$\forall$	For all
$\bigcup_{n=1}^{\infty}$	Infinite union
$\bigcap_{n=1}^{\infty}$	Infinite intersection
$n \to c$	n approaches c
ϕ	Empty set
$f(t) \leftrightarrow F(\omega)$	$F(\omega)$ is Fourier Transform of $f(t)$
$f(t) * g(t)$	Convolution of f and g
$\langle f, g \rangle$	Inner product of f and g
$F(f(t) * g(t))$	Fourier transform of a convolution

CHAPTER 1

INTRODUCTION

Digital Signal Processing (DSP) has been the driving force for the development of modern scientific technologies. Its applications have been seen from space technology to household electronics. An indispensable component of digital signal processing is *sampling* because an analog signal has to be sampled before any DSP procedures can be applied, see Fig. 1.1.

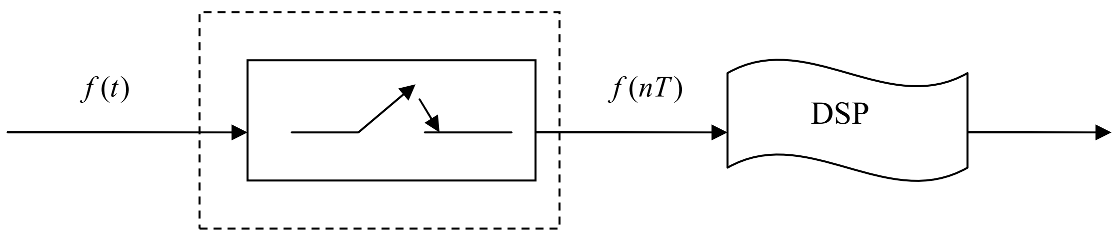

Figure 1.1 Sampling and Digital Signal Processing

The process of selecting values of an analog signal at discrete-time instants is called sampling. A sampling device is used to take measurements of the analog signal at a regular interval of time. The interval of time, or so-called sampling interval, the reciprocal of which is called sampling rate, has to be carefully selected so that the samples capture the characteristics of the original analog signal. A well-known industrial standard for sufficient sampling of an analog signal is that the sampling rate is at least twice the highest frequency, also known as the Nyquist rate [J. G. Proakis and D. G. Manolakis, 1996], of the signal. The challenge of effective sampling comes from signals with high frequency components because the sampling device has to perform at a much faster rate to cope with the high frequencies, and such sampling devices are expensive to make. As the modern technologies advance at a faster pace than ever, DSP technologies have to be at least one-step ahead. Particularly with the emergence of broad-band signals from space, commercial applications, and military, the highest frequency of which is usually at the range of mega-hertz or even giga-hertz, construction of advanced sampling mechanisms have become one of the most active research areas in digital signal processing. The goal of developing new sampling methods is always the same, that is, to reduce the sampling rate to a level that is lower than the Nyquist rate.

In the past decade or so, higher-order sampling methods were developed to lower the sampling rate [Moon, 2000, Mitra, *et. al*., 1993, Xiao, 1995]. The idea of using a guard-band to reduce the susceptibility of the permissible sampling rates to aliasing was introduced in [Vaughan, *et. al*., 1991]. Similar treatment of the problem can be found in [Gaskell, 1978; Gregg, 1979; Coulson, 1994]. However, due to the excessive structural complexity, the implementation of those higher-order methods may add more cost to the making of such sampling devices.

It is observed that most broad-band signals are passband signals with existence of significant gaps among the spectral components of the signal displayed from the frequency domain [Proakis and Manolakis, 1996], see Fig.1.2. The objective of this research is to develop low-cost sampling methods to achieve a lower sampling rate than the Nyquist rate by utilizing the gaps among the spectra. The proposed sampling mechanism is guaranteed to be low-cost because only first-order sampling is considered, which has the simplest design structure, known as sample-and-hold. The idea of deriving such new sampling rates, both optimal and admissible sampling rates, is that the spectra of a passband signal do not intersect with each other (anti-aliasing) as they shift horizontally at a step size that is equal to the designated sampling rate.

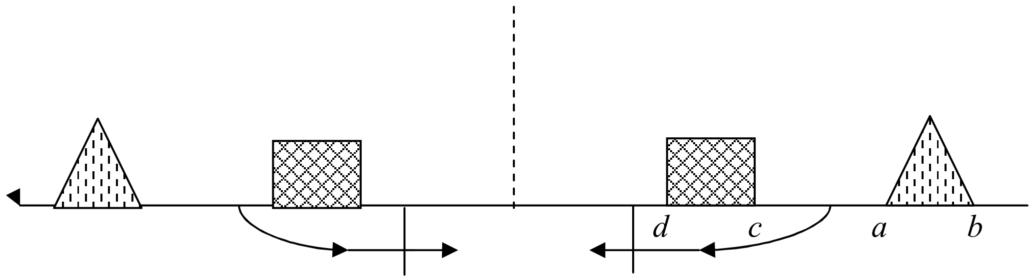

Figure 1.2 Two-Band passband signals with significant gaps

The Fig. 1.1 describes the continuous signal $f(t)$ being sampled by some process at regular intervals. The first box represents the sampling device. This process converts the continuous or analog signal to a digital one where the digital signal processor (DSP), manipulates and recreates the original. This work provides all possible sampling rates and determines the smallest. This improvement allows the sampler to perform at a much slower rate to recreate the original signal, hence saving money on sampler maintenance.

The idea of signal processing involves transforming the time dependent signal $f(t)$ into a frequency dependent function $F(\omega)$ via a Fourier transform, with sampling frequency $\Delta_f = \frac{1}{T}$. This transformation simplifies analysis from an often times difficult signal function to an analysis of a much easier one in frequency domain. Finding the smallest sampling rate translates into an increase in T, where T corresponds to the time delay in the sampling device. Allowing this delay to be as large as possible means the sampling device can sample much more slowly. One might ask what a feasible sampling rate is and what the industry standards are. It helps to observe the following diagram.

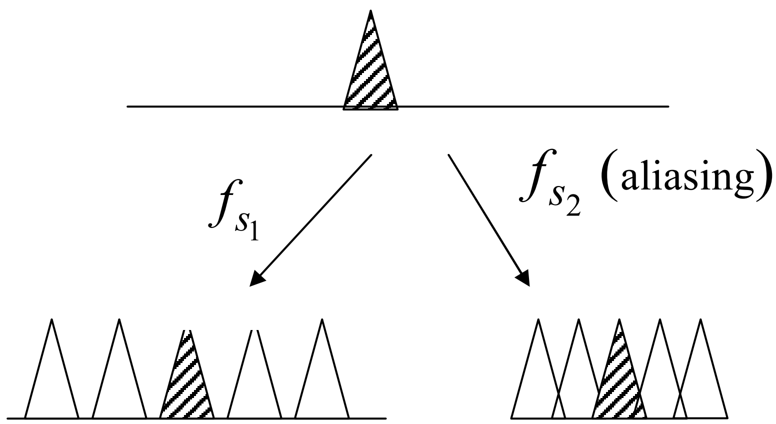

Figure 1.3 Sufficient vs. insufficient sampling

The Fig. 1.3 is an illustration of the original signal after conversion into frequency domain. After a signal has been sampled, copies of the signal in frequency domain are distributed along the frequency axis. The picture at the bottom left reflects a sampling frequency that is feasible. This frequency is considered feasible because no intersection occurs between copies. The picture at the bottom right reflects a sampling scheme that is not feasible. Copies using this frequency are observed to have intersections. This phenomenon is called aliasing and does not allow for successful reconstruction of the original signal. The industry standard for sampling is that the sampling rate be at least twice the bandwidth of the signal.

Using this standard would translate into, $\Delta_f > 2b$, with b being the highest frequency of the signal, in the case where the signal is in frequency domain. However, this sampling rate does not use valuable space between the spectra. Modifying the sampling rate to make use of these spaces can dramatically decrease the rate that a sampling machine has to sample the signal. In chapter 2, all possible sampling frequencies are determined with minimum constraints.

In what follows, we give a brief review of some basic facts in digital signal processing. Let $f(t)$ be an analog signal of time, then the Fourier transform into frequency domain takes the form

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}dt \tag{1-1}$$

Then the inversion formula will express $f(t)$ using $F(\omega)$ in the following way

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}d\omega \tag{1-2}$$

From analysis it is known that $\left\{e^{in\omega_0 t}\right\}_{n=-\infty}^{\infty}$ forms an orthogonal basis for periodic functions in L^2, with period T, and $\omega_0 = \frac{2\pi}{T}$. Then $f(t) = \sum_{n=-\infty}^{\infty} a_n e^{in\omega_0 t}$, implying,

$$\left\langle f(t), e^{im\omega_0 t}\right\rangle = \left\langle \sum_{n=-\infty}^{\infty} a_n e^{in\omega_0 t}, e^{im\omega_0 t}\right\rangle = a_m \int_{-\frac{T}{2}}^{\frac{T}{2}} e^{in\omega_0 t} e^{-im\omega_0 t}dt = a_m \int_{-\frac{T}{2}}^{\frac{T}{2}} e^{im\omega_0 t} e^{-im\omega_0 t}dt = a_m T .$$

Then, $a_m = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} f(t)e^{im\omega_0 t}dt$, from the orthogonality of the basis functions.

Combining the previous derivations yield the familiar Fourier series representation as:

$$f(t) = \sum_{n=-\infty}^{\infty} a_n e^{in\omega_0 t} \text{, with } a_n = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} f(t)e^{in\omega_0 t}dt \tag{1-3}$$

Introducing the Dirac delta function that is defined to satisfy the following conditions:

(i) $\delta(t) = 0$ when $t \neq 0$, and $\int_{-\infty}^{\infty} \delta(t)dt = 1$

(ii) $\delta(t) = \infty$ when $t = 0$ and the important property $\int_{-\infty}^{\infty} f(t)\delta(t)dt = f(0)$

From this definition, it is easy to verify the convolution $f * \delta = \int_{-\infty}^{\infty} f(\tau)\delta(t-\tau)d\tau = f(t)$.

Next, we introduce the impulse train function, $\mu(t) = \sum_{n=-\infty}^{\infty}\delta(t+nT)$ as periodic with period T. The Fourier series representation of $\mu(t)$ is,

$$\mu(t) = \sum_{n=-\infty}^{\infty} a_n e^{in\omega_0 t} \quad \text{, where } a_n = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} \mu(t)e^{-in\omega_0 t}dt .$$

Therefore,

$$a_n = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} \sum_{n=-\infty}^{\infty}\delta(t+nT)e^{-in\omega_0 t}dt .$$

Since $t \in \left[-\frac{T}{2}, \frac{T}{2}\right]$, $a_n = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} \delta(t)dt = \frac{1}{T}$.

It is known that the Fourier transform of two convoluted functions is the product of their Fourier transforms, that is,

$$f(t) * g(t) \leftrightarrow F(\omega)G(\omega) . \tag{1-4}$$

To show that $\frac{1}{T}\sum_{n=-\infty}^{\infty} e^{in\omega_0 t} \leftrightarrow \frac{2\pi}{T}\sum_{n=-\infty}^{\infty}\delta(\omega - n\omega_0)$, one may use the following derivation:

$$F^{-1}\{\frac{1}{T}\sum_{n=-\infty}^{\infty}\delta(\omega-n\omega_0)\} = \frac{1}{2\pi}\int_{-\infty}^{\infty}\frac{2\pi}{T}\sum_{n=-\infty}^{\infty}\delta(\omega-n\omega_0)e^{i\omega t}d\omega = \frac{1}{T}\sum_{n=-\infty}^{\infty}\int_{-\infty}^{\infty}\delta(\omega-n\omega_0)e^{i\omega t}d\omega$$

$$= \frac{1}{T}\sum_{n=-\infty}^{\infty} e^{in\omega_0 t} .$$

Thus,

$$F\{f(t)*\frac{1}{T}\sum_{n=-\infty}^{\infty}e^{in\omega_0 t}\} = \frac{2\pi}{T}F(\omega)\sum_{n=-\infty}^{\infty}\delta(\omega-n\omega_0),$$

taking the inverse Fourier transform on both sides of the above equation, we obtain the following:

$$f(t)*\frac{1}{T}\sum_{n=-\infty}^{\infty}e^{in\omega_0 t} = \frac{1}{T}\int_{-\infty}^{\infty}F(\omega)\sum_{n=-\infty}^{\infty}\delta(\omega-n\omega_0)e^{i\omega t}d\omega = \frac{1}{T}\sum_{n=-\infty}^{\infty}\int_{-\infty}^{\infty}F(\omega)\delta(\omega-n\omega_0)e^{i\omega t}d\omega .$$

Again, using the properties of Dirac delta, the above derivation leads to

$$f(t)*\frac{1}{T}\sum_{n=-\infty}^{\infty}e^{in\omega_0 t} = \frac{1}{T}\sum_{n=-\infty}^{\infty}F(n\omega_0)e^{in\omega_0 t} .$$

Evaluating the convolution on the left gives

$$\sum_{n=-\infty}^{\infty}f(t+nT) = \frac{1}{T}\sum_{n=-\infty}^{\infty}F(n\omega_0)e^{in\omega_0 t} .$$

Introducing the functions $\overline{f(t)} = \sum_{n=-\infty}^{\infty}f(t+nT)$ and $\overline{F(\omega)} = \sum_{n=-\infty}^{\infty}F(\omega+n\omega_1)$, where $\omega_1 = \frac{2\pi}{T_1}$, it can be shown that

$$\overline{F(\omega)} = \frac{2\pi}{\omega_1}\sum_{n=-\infty}^{\infty}f(nT_1)e^{-inT_1\omega} \text{ with } T_1 = \frac{2\pi}{\omega_1}, \tag{1-5}$$

this is known as the Poisson Summation Formula. The outline of the derivation is stated as follows: we begin with a series representation for the periodic impulse train in the frequency domain:

$$\sum_{n=-\infty}^{\infty}\delta(\omega+n\omega_1) = \sum_{n=-\infty}^{\infty}a_n e^{-inT_1\omega} ,$$

and a_n can be easily computed to be $\frac{1}{\omega_1}$. Thus,

$$F(\omega) * \sum_{n=-\infty}^{\infty} \delta(\omega + n\omega_1) = F(\omega) * \frac{1}{\omega_1} \sum_{n=-\infty}^{\infty} e^{-inT_1\omega} \tag{1-6}$$

evaluating the left side of (1-6) gives

$$\int_{-\infty}^{\infty} F(\tau) \sum_{n=-\infty}^{\infty} \delta(\tau - (\omega + n\omega_1)) d\tau = \sum_{n=-\infty}^{\infty} \int_{-\infty}^{\infty} F(\tau)\delta(\tau - (\omega + n\omega_1)) d\tau .$$

Using the properties of Dirac delta, the integral on the left of (1-6) becomes $\sum_{n=-\infty}^{\infty} F(\omega + n\omega_1)$. Evaluating the convolution on the right side of (1-6) gives

$$\int_{-\infty}^{\infty} F(\tau) \frac{1}{\omega_1} \sum_{n=-\infty}^{\infty} e^{-inT_1(\omega-\tau)} d\tau = \frac{1}{\omega_1} \sum_{n=-\infty}^{\infty} \int_{-\infty}^{\infty} F(\tau) e^{-inT_1(\omega-\tau)} d\tau = \frac{1}{\omega_1} \sum_{n=-\infty}^{\infty} e^{-inT_1\omega} \int_{-\infty}^{\infty} F(\tau) e^{inT_1\tau} d\tau$$

$$= \frac{2\pi}{\omega_1} \sum_{n=-\infty}^{\infty} f(nT_1) e^{-inT_1\omega} .$$

Combining the computation of both sides of the (1-6) produces Poisson's Summation Formula. $\overline{F(\omega)} = \frac{2\pi}{\omega_1} \sum_{n=-\infty}^{\infty} f(nT_1) e^{-inT_1\omega}$ with $T_1 = \frac{2\pi}{\omega_1}$.

The goal of these calculations is to express the original signal in terms of its samples. The Poisson formula is close but not quite there. Taking inverse Fourier transforms would not produce the original signal. In order to achieve the goal, we must first introduce an ideal low pass filter and then present the famous Shannon's Sampling Theorem [J. G. Proakis and D. G. Manolakis, 1996]. The Fourier Transform of the function

$$f_\delta(t) = \sum_{n=-\infty}^{\infty} Tf(nT)\delta(t - nT) = \sum_{n=-\infty}^{\infty} Tf(nT) e^{-in\omega t}$$

and together with (1-5) yield $\sum_{n=-\infty}^{\infty} F(\omega + n\omega_0)$. Noticing that this process has created multiple copies of the Fourier Transform of $f(t)$ in frequency domain, it is suitable to design a filter to extract only the original frequency domain signals from their copies.

Consider the ideal lowpass filter, $S(\omega) = 1$ if $|\omega| < \sigma$ and 0 otherwise. Taking the inverse Fourier Transform of $S(\omega)$ gives

$$s(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S(\omega) e^{i\omega t} d\omega = \frac{1}{2\pi} \int_{-\sigma}^{\sigma} e^{i\omega t} d\omega = \frac{1}{\pi t} \sin(\sigma t).$$

From the definition of the lowpass filter, $S(\omega)$, it is clear that

$$S(\omega) \sum_{n=-\infty}^{\infty} F(\omega + n\omega_0) = F(\omega) \text{ whenever } \frac{1}{T} \geq \frac{\sigma}{\pi}. \tag{1-7}$$

The minimum sampling rate 2σ is known as the Nyquist rate, and it specifies a sampling rate to guarantee perfect reconstruction of a signal from the samples. Taking the inverse Fourier Transform of both sides of (1-7) gives $s(t) * f_\delta(t) = f(t)$, which leads to

$$f(t) = \sum_{n=-\infty}^{\infty} \frac{f(nT) \sin \sigma(t - nT)}{\sigma(t - nT)}. \tag{1-8}$$

This is the famous Shannon's Sampling Theorem. It proves that a signal $f(t)$ can be perfectly reconstructed from its samples with an ideal lowpass filter, given the Nyquist rate is satisfied.

We have now established that a signal can be reconstructed in this way given that the sampling rate is at least the Nyquist rate. The Nyquist rate guarantees that the copies of the signal in frequency domain do not overlap and avoid aliasing. However, in the case of a two-band passband signal, the Nyquist rate can be improved upon by allowing the copies of the spectra to move between each other. The proceeding work establishes the necessary and sufficient conditions for permissible and optimal sampling rates in the case of two-band passband signals.

CHAPTER 2

OPTIMAL AND PERMISSIBLE STEP SIZES FOR TWO DISJOINT INTERVALS

Before the discussion of necessary and sufficient conditions for the feasible step sizes, we introduce several important symbols to make the work transparent.

Shifted Interval

The symbol $I_{[a,b]}^{m\Delta}$ will designate the closed interval $[a+m\Delta, b+m\Delta]$, where $m \in \mathrm{N}$, $I_{[a,b]} = [a, b]$, and $\Delta \in R^{+}$.

Feasible Step Size

Throughout the text, it is always assumed that the intervals $[a,b]$ and $[d,c]$ are closed, disjoint, and satisfy $d < c < a < b$.

The symbol Δ_f will be used to designate a feasible step size. Δ_f is a feasible step size for two intervals $[a,b]$ and $[d,c]$ if,

i.) $I_{[a,b]}^{m\Delta}$ intersects $I_{[d,c]}^{n\Delta}$ at only one point, or the intersection is empty, for all non-negative integers m and n.

ii.) $I_{[a,b]}^{m\Delta}$ intersects $I_{[a,b]}^{n\Delta}$ at only one point, or the intersection is empty, for all non-negative integers m and n.

iii.) $I_{[d,c]}^{m\Delta}$ intersects $I_{[d,c]}^{n\Delta}$ at only one point, or the intersection is empty, for all non-negative integers m and n.

Optimal Step Size

The symbol Δ_o will be used to designate the optimal step size. Δ_o is optimal if all $\Delta_f \geq \Delta_o$.

The Floor Function

The symbol $\lfloor x \rfloor$ denotes the greatest integer less than x. For example, $\lfloor 3.14 \rfloor$ =3.

The following Propositions (P1-P4) determine feasible step sizes for two arbitrary intervals illustrated in the following diagram:

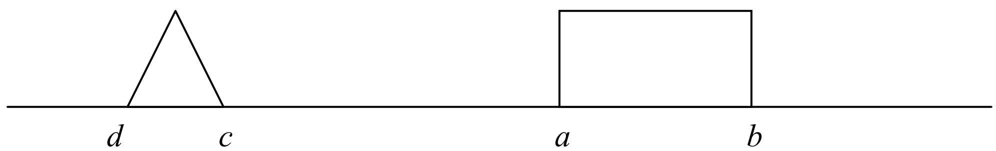

Figure 2.1 Arbitrary two-band

Proposition 2.1 If $\Delta \geq (b-a)$, then $I_{[a,b]}^{m\Delta} \cap I_{[a,b]}^{n\Delta} = \phi$ or a singleton for integers m and n.

Proof:

This is the Nyquist rate and is known.

Proposition 2.2 Let $[d,c]$ and $[a,b]$ be closed disjoint intervals; if $\exists m \in Z^{+}$ such that $c + m\Delta \leq a$ and $d + (m+1)\Delta \geq b$, then $I_{[d,c]}^{n\Delta} \cap I_{[a,b]}$ is empty or a singleton for all non-negative integers n .

Proof:

The proof is elementary. Any integer less than m would clearly not cause an intersection as well as any integer greater than m + 1. The integers m and m + 1 cause at most one intersection namely at the points a or b respectively.

Proposition 2.3 Let $[a,b]$and $[d,c]$be closed disjoint intervals on the real line. If $I_{[d,c]}^{m\Delta} \cap I_{[a,b]} = \phi$ or a singleton for all integers m, then $I_{[d,c]}^{m\Delta} \cap I_{[a,b]}^{n\Delta} = \phi$ or a singleton for all integers m, n.

Proof:

The proof is by contradiction. Let $I^{m\Delta}_{[d,c]} \cap I_{[a,b]} = \phi$ or a singleton hold for all $m \in N$ and pick k, $p \in N$ such that $I^{k\Delta}_{[d,c]} \cap I^{p\Delta}_{[a,b]}$ contains more than a point; then clearly k must be greater than p for a non-empty intersection to occur.

Case 1

$d + k\Delta \leq a + p\Delta < c + k\Delta \leq b + p\Delta$. Subtracting $p\Delta$ from each of the four expressions

$d + (k-p)\Delta \leq a < c + (k-p)\Delta \leq b$, which contradicts $I^{m\Delta}_{[d,c]} \cap I_{[a,b]} = \phi$ or a singleton.

Case 2

$d + k\Delta \leq a + p\Delta < b + p\Delta \leq c + k\Delta$. Subtracting $p\Delta$ from each of the four expressions yields

$d + (k-p)\Delta \leq a < b \leq c + (k-p)\Delta$, which contradicts $I^{m\Delta}_{[d,c]} \cap I_{[a,b]} = \phi$ or a singleton.

Case 3

$a + p\Delta < d + k\Delta \leq b + p\Delta \leq c + k\Delta$. Subtracting $p\Delta$ again from each expression

yields

$a < d + (k-p)\Delta \leq b \leq c + (k-p)\Delta$, contradicting $I^{m\Delta}_{[d,c]} \cap I_{[a,b]} = \phi$ or a singleton.

Case 4

$a + p\Delta \leq d + k\Delta < c + k\Delta \leq b + p\Delta$. Subtracting $p\Delta$ yields

$a \le d + (k-p)\Delta < c + (k-p)\Delta \le b$, again contradicting $I_{[d,c]}^{m\Delta} \cap I_{[a,b]} = \phi$ or a singleton. Since all four cases cover any intersection and each lead to a contradiction, the proposition must hold.

Proposition 2.4 Let $[d,c]$ and $[a,b]$ be closed disjoint intervals such that $\Delta \ge (c-d)$ and $\Delta \ge (b-a)$. If $\exists \ m \in Z^{+}$ such that $c + m\Delta \le a$ and $d + (m+1)\Delta \ge b$, then Δ is a feasible step size.

Proof:

Combining Proposition 2.1, 2.2, and 2.3 implies the statement.

The next set of propositions (P5-P14) will allow us to partition the positive real line by using a set of points calculated from the integer $t = \left\lfloor \frac{a-c}{(b-a)+(c-d)} \right\rfloor$. This value can be thought of as the capacity of the gap between the two intervals $[a,b]$ and $[d,c]$ to accommodate the intervals shifting through the gap without causing intersections. Using this value, we can partition the positive side of the real line and determine feasibility of the stepsize.

Proposition 2.5 If $t = \left\lfloor \frac{a-c}{(b-a)+(c-d)} \right\rfloor$, then $\frac{b-d}{t+1} \le \frac{a-c}{t}$.

Proof:

By definition, $t \le \frac{a-c}{(b-a)+(c-d)}$. Because the quantity is clearly positive, we have

$$\frac{1}{t} \ge \frac{(b-a)+(c-d)}{a-c} = \frac{(b-d)-(a-c)}{a-c} = \frac{b-d}{a-c} - 1, \text{ then}$$

$$\frac{t+1}{t} \ge \frac{b-d}{a-c} \quad \text{implying that } \frac{b-d}{t+1} \le \frac{a-c}{t}.$$

Proposition 2.6 Let $t = \left\lfloor \frac{a-c}{(b-a)+(c-d)} \right\rfloor$; then $\frac{b-d}{t+1} \geq (b-a)$ and $\frac{b-d}{t+1} \geq (c-d)$

Proof:

The above inequalities can be obtained from,

$$\frac{b-d}{t+1} \geq \frac{b-d}{\frac{a-c}{(b-a)+(c-d)}+1} = \frac{b-d}{\frac{b-d}{(b-a)+(c-d)}} = (b-a)+(c-d).$$

Proposition 2.7 Let $t = \left\lfloor \frac{a-c}{(b-a)+(c-d)} \right\rfloor$; then $\frac{b-d}{t+1-n} \leq \frac{a-c}{t-n}$ for $n = 0 \ldots . t-1$

Proof:

By definition, $t \leq \frac{a-c}{(b-a)+(c-d)}$ which implies $t - n \leq \frac{a-c}{(b-a)+(c-d)}$

Using the same argument as above we obtain

$$\frac{1}{t-n} \geq \frac{a-c}{(b-a)+(c-d)} = \frac{b-d}{a-c} - 1,$$

and $\frac{t+1-n}{t-n} \geq \frac{b-d}{a-c}$, which implies $\frac{b-d}{t+1-n} \leq \frac{a-c}{t-n}$.

Thus the sequence $\left\{ \left[\frac{b-d}{t+1-n}, \frac{a-c}{t-n} \right] \right\}_{n=0}^{t-1}$ forms a sequence of closed disjoint intervals, because $(a-c) < (b-\mathrm{d})$ implying $\frac{a-c}{t-n} < \frac{b-d}{t-n}$. The fact that the intervals are disjoint will be crucial to the development of a fast algorithm to determine intersections later in the work.

Proposition 2.8 Let $t = \left\lfloor \dfrac{a-c}{(b-a)+(c-d)} \right\rfloor$; then $\dfrac{b-d}{t+1+n} > \dfrac{a-c}{t+n}$ for n = 1, 2……..

Proof:

Since $n \geq 1$, we have $t + n > \dfrac{a-c}{(b-a)+(c-d)}$.

Thus,

$$\frac{1}{t+n} < \frac{(b-a)+(c-d)}{a-c} = \frac{b-d}{a-c} - 1,$$

which implies

$$\frac{t+1+n}{t+n} < \frac{b-d}{a-c}. \text{ Hence, } \frac{b-d}{t+1+n} > \frac{a-c}{t+n}.$$

Proposition 2.9 If $t = \left\lfloor \dfrac{a-c}{(b-a)+(c-d)} \right\rfloor$, then $\displaystyle\bigcup_{n=1}^{\infty}\left(\frac{a-c}{t+n}, \frac{b-d}{t+n}\right) = \left(0, \frac{b-d}{t+1}\right)$.

Proof:

Let $x \in \displaystyle\bigcup_{n=1}^{\infty}\left(\frac{a-c}{t+n}, \frac{b-d}{t+n}\right)$, then $x \in \left(\dfrac{a-c}{t+k}, \dfrac{b-d}{t+k}\right)$ for some k. Then

$0 < \dfrac{a-c}{t+k} < x < \dfrac{b-d}{t+k} \leq \dfrac{b-d}{t+1}$ implying that $x \in \left(0, \dfrac{b-d}{t+1}\right)$.

Let $x \in \left(0, \dfrac{b-d}{t+1}\right)$, then $0 < x < \dfrac{b-d}{t+1}$, so $\exists \varepsilon > 0$ such that $x > \varepsilon$.

Since $\dfrac{a-c}{t+n} \to 0$ as $n \to \infty$, $\forall \varepsilon > 0 \quad \exists$ N : $n >$ N implies $\dfrac{a-c}{t+n} < \varepsilon$, thus $\dfrac{a-c}{t+n} < x$. Let M denote the smallest positive integer such that $\dfrac{a-c}{t+M} < x$, (Well-Ordering Property). Thus, $\dfrac{a-c}{t+M} < x \leq \dfrac{a-c}{t+M-1}$. If M > 1, from Proposition

2.8 we obtain, $\frac{a-c}{t+M-1} < \frac{b-d}{t+M}$ yielding $\frac{a-c}{t+M} < x \leq \frac{a-c}{t+M-1} < \frac{b-d}{t+M}$. If $M = 1$, and $x \in \left(0, \frac{b-d}{t+1}\right)$ then $\frac{a-c}{t+1} < x < \frac{b-d}{t+1}$ then $x \in \left(\frac{a-c}{t+1}, \frac{b-d}{t+1}\right)$ proving the proposition.

Proposition 2.10 Let $\Delta_x \in \left(0, \frac{b-d}{t+1}\right)$ where $t = \left\lfloor \frac{a-c}{(b-a)+(c-d)} \right\rfloor$; then Δ_x is not a feasible solution.

Proof:

Since $\Delta_x \in \left(0, \frac{b-d}{t+1}\right)$, then by Proposition 2.9 $\Delta_x \in \bigcup_{n=1}^{\infty} \left(\frac{a-c}{t+n}, \frac{b-d}{t+n}\right)$.

Therefore, $\Delta_x \in \left(\frac{a-c}{t+k}, \frac{b-d}{t+k}\right)$ for some integer $k \geq 1$. Then,

$\frac{a-c}{t+k} < \Delta_x < \frac{b-d}{t+k}$, implying $c + (t+k)\Delta_x > a$ and $d + (t+k)\Delta_x < b$

Case 1

If $c + (t+k)\Delta_x \geq b$, then $d + (t+k)\Delta_x < b \leq c + (t+k)\Delta_x$, intersecting in an interval to the left of b.

Case 2

If $c + (t+k)\Delta_x < b$, then $a < c + (t+k)\Delta_x < b$, intersecting around $c + (t+k)\Delta_x$. Since both intersections contain more than just a singleton, Δ_x is not a feasible solution.

Proposition 2.11 Let $\Delta_x \in \bigcup_{n=0}^{t-1} \left(\frac{a-c}{t-n}, \frac{b-d}{t-n}\right)$ and $t = \left\lfloor \frac{a-c}{(b-a)+(c-d)} \right\rfloor$; then Δ_x is not a feasible step size.

Proof:

If $\Delta_x \in \bigcup_{n=0}^{t-1} \left(\frac{a-c}{t-n}, \frac{b-d}{t-n}\right)$, then $\Delta_x \in \left(\frac{a-c}{t-k}, \frac{b-d}{t-k}\right)$ for some k less than $t-1$, which implies $c + (t-k)\,\Delta_x > a$ and $d + (t-k)\,\Delta_x < b$

Case 1

If $c + (t-k)\,\Delta_x \geq b$, then $d + (t-k)\,\Delta_x < b \leq c + (t-k)\,\Delta_x$, intersecting in an interval to the left of b.

Case 2

If $c + (t-k)\,\Delta_x < b$, then $a < c + (t-k)\,\Delta_x < b$, intersecting around $c + (t-k)\,\Delta_x$.

Thus, Δ_x is not a feasible step size.

Proposition 2.12 Let $\Delta_f \in \bigcup_{n=1}^{t-1} \left[\frac{b-d}{t-n+1}, \frac{a-c}{t-n}\right] \bigcup [b-d, \infty)$, $t = \left\lfloor \frac{a-c}{(b-a)+(c-d)} \right\rfloor$, then Δ_f is a feasible stepsize.

Proof:

If $\Delta_f \in [b-d, \infty)$, then it is greater than the Nyquist rate, and is feasible. Thus, we only consider,

$\Delta_f \in \bigcup_{n=1}^{t-1} \left[\frac{b-d}{t-n+1}, \frac{a-c}{t-n}\right]$. Then $\Delta_f \in \left[\frac{b-d}{t-k+1}, \frac{a-c}{t-k}\right]$ for some k = 1, 2, ……. t -1,

implying $\frac{b-d}{t-k+1} \leq \Delta_f \leq \frac{a-c}{t-k}$. Thus $c + (t-k)\,\Delta_f \leq a$ and

$d + (t - k + 1)\,\Delta_f \geq b$. From Proposition 2.4 and 2.6, Δ_f is a feasible solution with $\frac{b-d}{t+1}$ being the smallest feasible solution and hence the optimal solution.

Proposition 2.13 Let $t = \left\lfloor \frac{a-c}{(b-a)+(c-d)} \right\rfloor$; then

$$\bigcup_{n=0}^{k}\left[\frac{b-d}{t+1-n},\frac{a-c}{t-n}\right]\cup\left(\frac{a-c}{t-n},\frac{b-d}{t-n}\right) = \left[\frac{b-d}{t+1},\frac{b-d}{t-k}\right) \text{ for } k = 0 \ldots.\ t-1.$$

Proof:

The proof is by induction.

i.) If $k = 0$, $\left[\frac{b-d}{t+1},\frac{a-c}{t}\right]\cup\left(\frac{a-c}{t},\frac{b-d}{t}\right) = \left[\frac{b-d}{t+1},\frac{b-d}{t}\right)$.

ii.) Assume the statement is true for all integers less than k, then

$$\bigcup_{n=0}^{k}\left[\frac{b-d}{t+1-n},\frac{a-c}{t-n}\right]\cup\left(\frac{a-c}{t-n},\frac{b-d}{t-n}\right)$$

$$=\bigcup_{n=0}^{k-1}\left[\frac{b-d}{t+1-n},\frac{a-c}{t-n}\right]\cup\left(\frac{a-c}{t-n},\frac{b-d}{t-n}\right)\cup\left[\frac{b-d}{t+1-k},\frac{a-c}{t-k}\right]\cup\left(\frac{a-c}{t-k},\frac{b-d}{t-k}\right)$$

$$=\left[\frac{b-d}{t+1},\frac{b-d}{t-(k-1)}\right)\cup\left[\frac{b-d}{t+1-k},\frac{a-c}{t-k}\right]\cup\left(\frac{a-c}{t-k},\frac{b-d}{t-k}\right)$$

$$=\left[\frac{b-d}{t+1},\frac{a-c}{t-k}\right]\cup\left(\frac{a-c}{t-k},\frac{b-d}{t-k}\right)$$

$$=\left[\frac{b-d}{t+1},\frac{b-d}{t-k}\right).$$

Proposition 2.14 Let $t = \left\lfloor \frac{a-c}{(b-a)+(c-d)} \right\rfloor$; then

$$\bigcup_{n=0}^{t-1}\left[\frac{b-d}{t+1-n},\frac{a-c}{t-n}\right]\cup\left(\frac{a-c}{t-n},\frac{b-d}{t-n}\right) = \left[\frac{b-d}{t+1},b-d\right).$$

Proof:

The result follows directly from Proposition 13.

Now, we are ready to prove the main result of the chapter. The preceding propositions have allowed us to decompose the positive real line into regions in which feasibility is known. Proposition 10 has shown that no feasible step sizes exist in the interval $\left(0, \frac{b-d}{t+1}\right)$. Proposition 14 partitions the interval $\left[\frac{b-d}{t+1}, b-d\right)$ into subintervals in which feasibility is known. Finally, the interval $[b-d, \infty)$, consists entirely of feasible step sizes. Observing Fig 2.1 will illustrate how the positive real line is decomposed into feasible and non-feasible regions.

Figure 2.2 Partition of the number line with the feasible and non-feasible intervals

We are ready to prove the feasible stepsize theorem.

Theorem 2.1 Δ_f is a feasible solution for the intervals $[d, c]$ and $[a, b]$ if and only if,

$$\Delta_f \in \bigcup_{n=0}^{t-1} \left[\frac{b-d}{t+1-n}, \frac{a-c}{t-n}\right] \cup [b-d, \infty), \text{ where } t = \left\lfloor \frac{a-c}{(b-a)+(c-d)} \right\rfloor.$$

Proof:

Since the interval $\left(0, \frac{b-d}{t+1}\right)$ contains no feasible step sizes by Proposition 10, and the union, $\bigcup_{n=0}^{t-1} \left(\frac{a-c}{t-n}, \frac{b-d}{t-n}\right)$ contains no feasible step sizes by Proposition 11, and

$\bigcup_{n=0}^{t-1}\left[\frac{b-d}{t+1-n},\frac{a-c}{t-n}\right]\cup\left(\frac{a-c}{t-n},\frac{b-d}{t-n}\right) = \left[\frac{b-d}{t+1},b-d\right)$, then all possible step sizes contained in $(0,b-d)$ must belong to the union $\bigcup_{n=1}^{t-1}\left[\frac{b-d}{t-n+1},\frac{a-c}{t-n}\right]\bigcup[b-d,\infty)$ by Proposition 2.12.

■

The following diagram illustrates the role of each proposition in the proof of the main theorems.

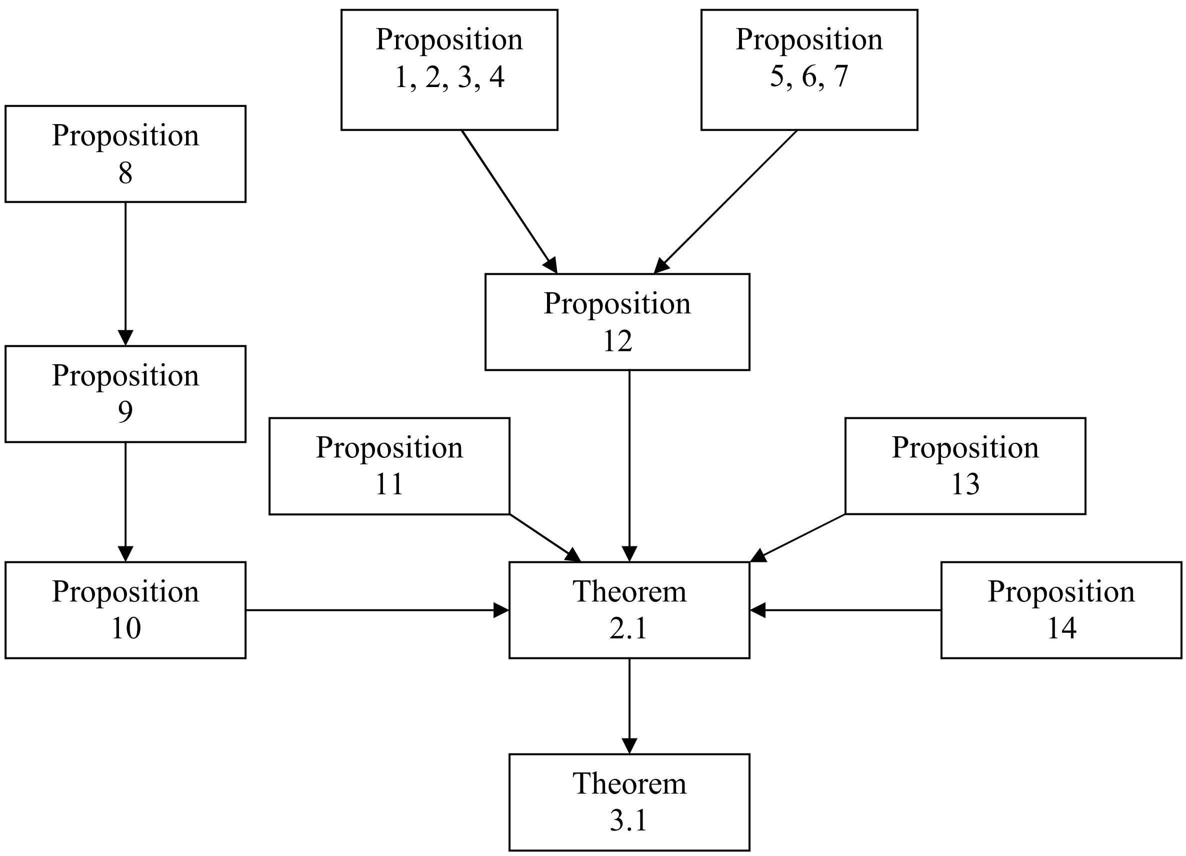

Figure 2.3 Flowchart for the relations among the propositions and the main theorems

The following is a simulation example for Theorem 2.1. We choose $[a,b]=[10,13]$, and $[d,c]=[-14,-10]$. According to Theorem 2.1, the feasible stepsizes are calculated and shown in the figure below.

The following diagram illustrates the new location of the blocks after a single shift with the optimal stepsize ($\Delta = 9$).

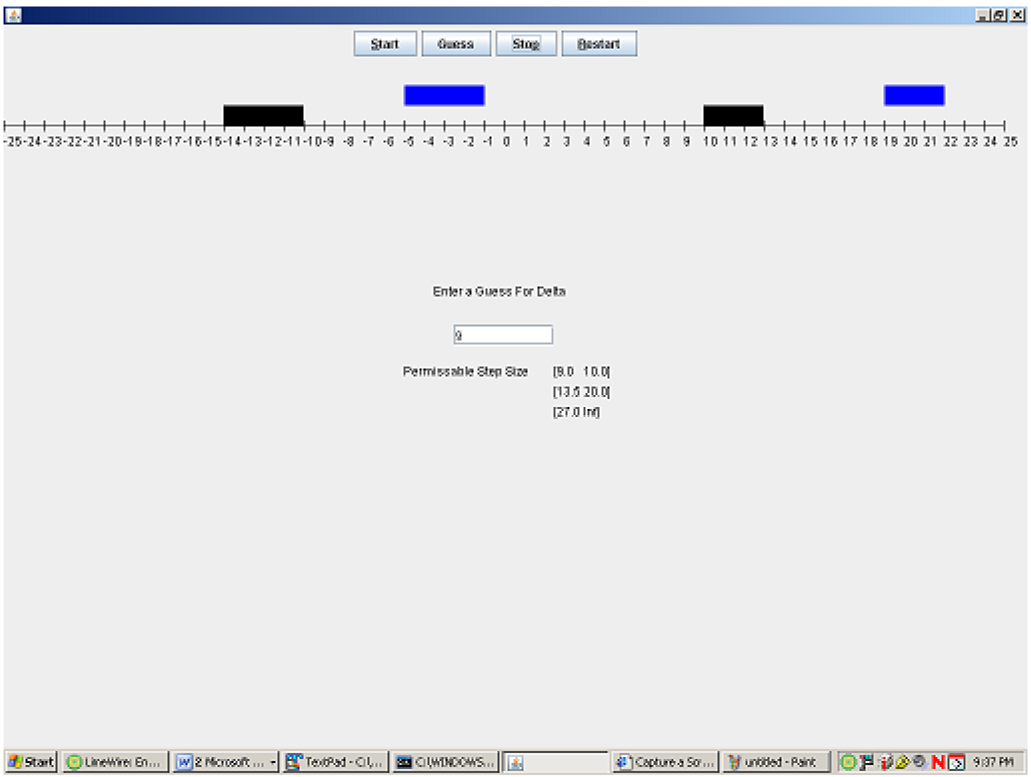

Figure 2.4 Shifting of two blocks after one step at the optimal stepsize

This diagram shows the new location of the blocks after two shifts, notice that the block is allowed to move inside the gap.

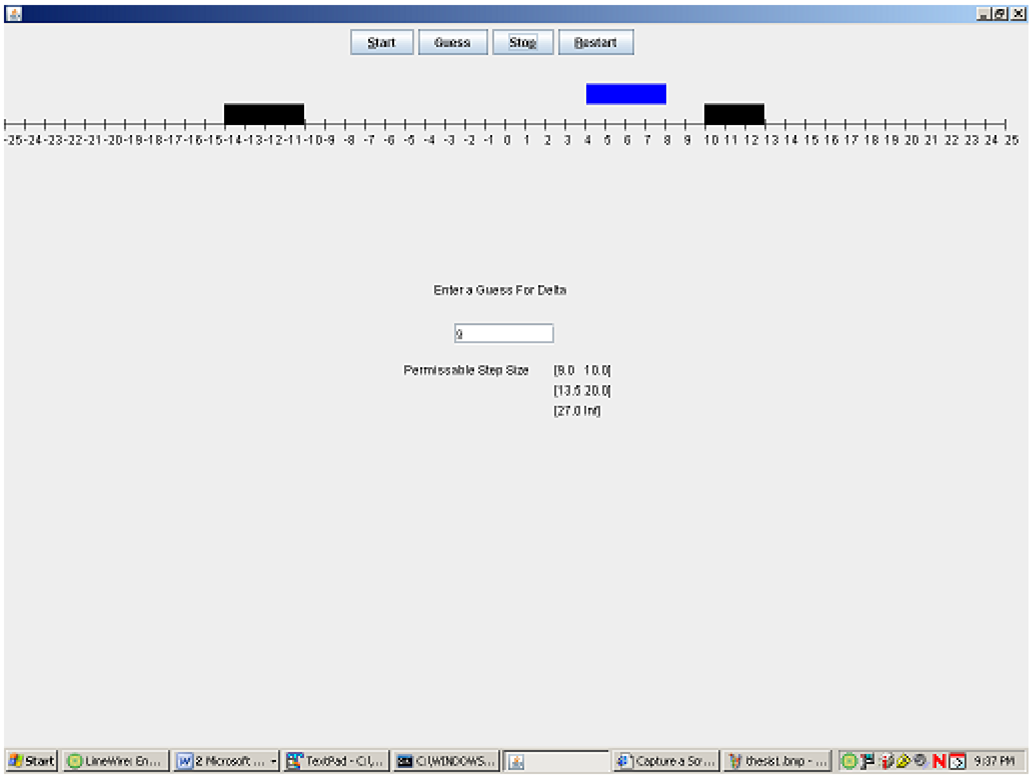

Figure 2.5 Shifting of two blocks after two steps at the optimal stepsize

This diagram illustrates the new location of the blocks, indicating that there is no intersection between the two blocks on the right.

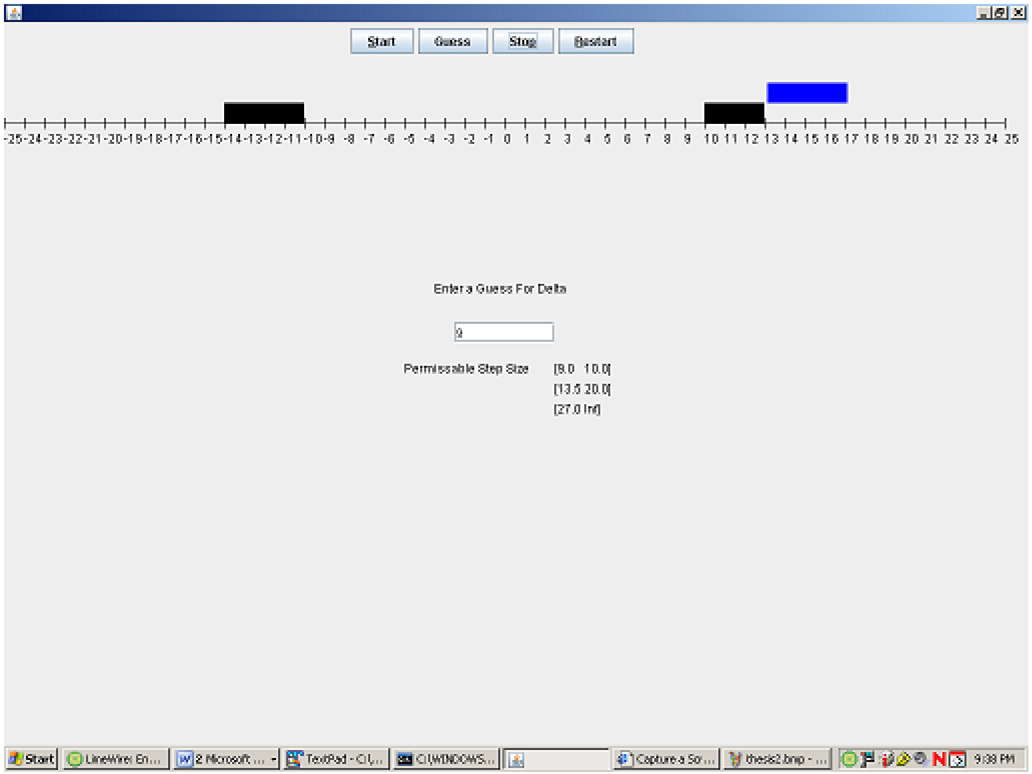

Figure 2.6 Shifting of two blocks after three steps at the optimal stepsize

CHAPTER 3

EXTENSION AND FAST ALGORITHM

Since all possible options for a feasible solution have been examined and the necessary and sufficient conditions for feasibility have been established for two disjoint closed intervals, we can extend the result to an arbitrary number of closed disjoint intervals by collecting feasible solutions from the pair-wise solutions and determining their intersections.

Pseudo-code

In this section, a suitable algorithm is developed to determine a set of feasible solutions in the case where multiple intervals occur and their shifted copies are not allowed to intersect. The primary advantage of the algorithm is that it is not computationally demanding. The relatively small complexity is strongly dependent on the fact that the set of all feasible solutions for each pair of intervals is a union of disjoint sets. Another key observation is that every one of the intervals has a lower bound greater than the upper bound of the previous, that is $(a - c) < (b - d)$ or $\frac{a-c}{t-n} < \frac{b-d}{t-n}$, from the discussions earlier in the work. The algorithm is developed as follows.

Initialization

Let I_{1,n_1}, I_{2,n_2} ....I_{k,n_k} each be a union of closed disjoint intervals with the property mentioned above that is the upper bound of an interval is strictly less than the lower bound of the next , and I_{i,n_i} is the i^{th} such union containing n_i such intervals. Thus,

$$\mathrm{I}_{1,n_1} = [a_{1,1}, b_{1,1}] \cup [a_{2,1}, b_{2,1}] \cup .... [a_{n_1,1}, b_{n_1,1}]$$

$$\text{I}_{2,n_2} = [a_{1,2}, b_{1,2}] \cup [a_{2,2}, b_{2,2}] \cup \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots [a_{n_2,2}, b_{n_2,2}]$$

.

$$\text{I}_{k,n_k} = [a_{1,k}, b_{1,k}] \cup [a_{2,k}, b_{2,k}] \cup \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots [a_{n_k,k}, b_{n_k,k}]$$

Step 1.

:if one of the unions is empty, then stop.

:find the maximum value of the left endpoint of the first interval in each union call it L_{max},

:find the minimum value of the right endpoint of the first interval in each union call it R_{min}.

Step 2.

: If $L_{max} \leq R_{min}$, then $[L_{max}, R_{min}]$ is an intersection and is stored in, F. Remove the interval that R_{min} occurred in from the union it was contained in, and go to Step 1.

:Else remove the interval that R_{min} occurred in from the union it was contained in and go to Step 1.

Correctness of the algorithm

Proposition 3.1 The algorithm finds the intersection of k such unions containing one interval each.

Proof:

Since each union contains only one interval, we are looking for the intersection of the k intervals.

i.) In the case of two intervals $[a_1, b_1]$ and $[a_2, b_2]$, there are four possible intersections.

Case 1.

$a_1 \le a_2 \le b_2 \le b_1$, in which the algorithm finds $[L_{\max}, R_{\min}]$ = $[a_2, b_2]$ to be the intersection.

Case 2.

$a_2 \le a_1 \le b_1 \le b_2$, in which the algorithm finds $[L_{\max}, R_{\min}]$ = $[a_1, b_1]$ to be the intersection.

Case 3.

$a_1 \le a_2 \le b_1 \le b_2$, in which the algorithm finds $[L_{\max}, R_{\min}]$ = $[a_2, b_1]$ to be the intersection.

Case 4.

$a_2 \le a_1 \le b_2 \le b_1$, in which the algorithm finds $[L_{\max}, R_{\min}]$ = $[a_1, b_2]$ to be the intersection.

ii.) Assume the algorithm finds the intersection of $k-1$ intervals for all non-negative integers less than k,

Then $\bigcap_{n=1}^{k}[a_n, b_n]$ = $\bigcap_{n=1}^{k-1}[a_n, b_n] \cap [a_k, b_k]$ = $[L_{\max}, R_{\min}] \cap [a_k, b_k]$. From case i the algorithm find the intersection of two intervals and hence the finite intersection of k intervals. From the principle of Mathematical Induction, the proposition holds.

Proposition 3.2 If the algorithm gives a non-empty intersection, $[L_{\max}, R_{\min}]$, for k intervals, then $[L_{\max}, R_{\min}]$ is indeed an intersection.

Proof:

Let $[a_i, b_i]$ be an arbitrary interval in the collection of k intervals, then from the algorithm we have $a_i \le L_{\max}$ and $b_i \le R_{\min}$. Since $L_{\max} \le R_{\min}$, $a_i \le L_{\max} \le R_{\min} \le b_i$, which gives the appropriate intersection.

The results from Proposition 3.1 and 3.2 assure us that the algorithm is reliable for finding the intersection of k intervals or that the algorithm will find all intersections and will not give nonintersecting solutions. But what about a sequence of unions? When we decide to find all feasible solutions for an arbitrary number of intervals we are left with finding intersections of sequence of unions, where each union represents a set of feasible solutions for two intervals. The fact that the two intervals generate a union in which each of the elements in the union, namely a closed interval, are pair-wise disjoint allows for removal of an interval from one union at each iteration of the algorithm, thus reducing the algorithm to one where only two linear searches are required at each iteration.

Proposition 3.3 If I_{1,n_1}, I_{2,n_2} ....I_{k,n_k} is a sequence of unions of closed disjoint intervals where the upper bound of an interval is strictly less than the lower bound of the next, then the algorithm finds the intersection, $\mathrm{I}_{1,n_1} \cap \mathrm{I}_{2,n_2} \cap \ldots\ldots\ldots\ldots \cap \mathrm{I}_{k,n_k}$.

Proof:

i.) From Proposition 3.1, the algorithm finds the intersection of the first k intervals of each collection if there is an intersection on the first iteration. The removal process removes an interval that can no longer cause intersection because each $a_{i+1} \ge b_i \ge \mathrm{R}_{\min}$ in every collection, thus the interval removed is pair-wise disjoint with each remaining interval.

ii.) Assume the algorithm has found the intersection if any on the n^{th} iteration with n intervals removed. Since each iteration produces an interval or the empty set, it is disjoint from one produced at a different iteration, the intersection is a set of disjoint intervals. Now, on the $(n + 1)^{st}$ iteration, the algorithm finds the intersection, if any, of the k intervals under consideration, (from Proposition 3.1). Since the interval removed is disjoint from the remaining intervals, all possible intersections involving those (n + 1) intervals have been considered. By the principles of Mathematical induction the process discovers all possible intersections involving the removed n intervals on the n^{th} iteration. From this result, and the fact that the process eventually terminates, the proposition must hold.

Complexity

The complexity of an algorithm is extremely important. If in this case the computational complexity of the algorithm was costly, it might not be beneficial to use a modified sampling scheme. The cost of determining an appropriate sampling rate would outweigh the cost of using the traditional Nyquist rate. However, the complexity proves to be relatively small compared with the current sampling rates. A worst case scenario is often helpful in setting a bound on the time an algorithm will take to achieve a task. In this situation, the worst case occurs when every interval except the last is removed from each of the unions.

Proposition 3.4 The complexity of the algorithm, C, satisfies

$$C \leq 2k\left(\sum_{i=1}^{k}(n_i - 1) + 1\right) \tag{3-1}$$

where k is number of unions and n_i are the number of intervals in the i^{th} union.

This observation follows from the algorithm performing two linear searches at each iteration on k elements which is of complexity $2k$, the maximum number of iterations the algorithm can perform $\left(\sum_{i=1}^{j}(n_i - 1) + 1\right)$ occurs when each of the unions contains only it's last interval, the last iteration produces the extra 1.

CHAPTER 4

TWO-BAND PASSBAND SAMPLING THEOREM

In chapter 2, all feasible step sizes for two arbitrary closed disjoint intervals. This chapter focuses on the two-band passband signal. Using Theorem 2.1 to determine all feasible step sizes for distinct pairs, then using the Algorithm to find the intersection will give all feasible step sizes for the two-band passband signal.

Definition 4.1 Let $I_{[d,c]\cup[a,b]} = [-c,-d]\cup[-b,-a]\cup[d,c]\cup[a,b]$. A signal $f(t)$ is said to be bandpassed to $[d,c]\cup[a,b]$, see Fig. 1.2, if its Fourier transform $F(\omega)$ satisfies the condition $F(\omega)=0$ if $\omega \notin I_{[d,c]\cup[a,b]}$.

The illustration in Fig 1.2 gives a general idea of a signal bandpassed to $[d,c]\cup[a,b]$. Two bands on the negative frequency axis are mirror images of the original two bands on the positive side. In this case, there are four distinct pairings to consider between bands. From Theorem 2.1, we calculate the feasible step sizes for each pair and use the Algorithm to calculate the feasible sampling rates from the six sets of step sizes.

Ideal Two-Band Filter Design

Now that we know the feasible sampling rates for a signal bandpassed to $[d,c]\cup[a,b]$, we need to design a filter to retrieve the original spectra in the frequency domain. We adopt the csinc-function, first introduced in [Y. Wu, 2005], for the ideal bandpass filters In that case

$$S(\omega)=\begin{cases}1 & \omega \in I_{[d,c]\cup[a,b]} \\ 0 & otherwise\end{cases} \tag{4-1}$$

The inverse Fourier Transform of $s(t)$ is

$$s(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty}S(\omega)e^{j\omega t}d\omega=\frac{1}{2\pi}2\left[\int_{d}^{c}\cos(\omega t)d\omega+\int_{a}^{b}\cos(\omega t)d\omega\right]$$

$$=\frac{1}{t\pi}\left[\sin(ct)-\sin(dt)+\sin(bt)-\sin(at)\right]$$

$$=\frac{1}{t\pi}\left[2\cos(\frac{c+d}{2})t\sin(\frac{c-d}{2})t+2\cos(\frac{a+b}{2})t\sin(\frac{b-a}{2})t\right].$$

Letting $\sigma_1=c-d$, $\omega_1=\frac{c+d}{2}$, $\sigma_2=b-a$, $\omega_2=\frac{a+b}{2}$, we obtain

$$s(t)=\frac{\sigma_1}{\pi}c\sin c_{[\omega_1,\sigma_1]}(t)+\frac{\sigma_2}{\pi}c\sin c_{[\omega_2,\sigma_2]}(t) \tag{4-2}$$

where $\mathrm{csinc}_{[\omega,\sigma]}(t)=\frac{\cos(\omega t)\sin(\sigma t/2)}{\sigma t/2}$. This filter will extract the original spectra from the copies.

Sampling Theorem for Two-Band Signals

Inspection of Fig 1.2 reveals that a two-band signal bandpassed over $[d,c]\cup[a,b]$, gives six pair-wise comparisons between spectra. We list all the comparisons: $[-b,-a]$ with $[-d,-c]$, $[-b,-a]$ with $[d,c]$, $[-b,-a]$ with $[a,b]$, $[-d,-c]$ with $[d,c]$, $[-d,-c]$ with $[a,b]$, $[d,c]$ with $[a,b]$. Using Theorem 2.1 on each comparison gives a set of feasible step sizes for each. Finally, using the algorithm will determine the intersection of each set of feasible step sizes.

Theorem 4.1 Suppose a signal $f(t)$ is bandpassed over $[d,c]\cup[a,b]$. Let ω_s and T be the sampling frequency and sampling interval, respectively, and $\omega_s=\frac{2\pi}{T}$. Then, $f(t)$ can be completely determined from its samples $f(nT)$ via

$$f(t)=\sum_{n=-\infty}^{\infty} f(nT)\left\{\frac{2\sigma_1}{\omega_s}\operatorname{csinc}_{[\omega_1,\sigma_1]}(t-NT)+\frac{2\sigma_2}{\omega_s}\operatorname{csinc}_{[\omega_2,\sigma_2]}(t-NT)\right\} \tag{4-3}$$

where, $\operatorname{csinc}_{[\omega,\sigma]}(t)=\dfrac{\cos(\omega t)\sin(\sigma t/2)}{\sigma t/2}$, $\sigma_1=c-d$, $\omega_1=\dfrac{c+d}{2}$, $\sigma_2=b-a$, and $\omega_2=\dfrac{a+b}{2}$, if and only if the sampling frequency ω_s satisfies the following feasibility condition,

$$\omega_s \in \bigcap_{k=1}^{6}\left\{\bigcup_{n=0}^{t_k-1}\left[\frac{b_k-d_k}{t_k+1-n},\frac{a_k-c_k}{t_k-n}\right]\cup\left[b_k-d_k,\infty\right)\right\} \tag{4-4}$$

where $t_k=\left\lfloor\dfrac{a_k-c_k}{(b_k-a_k)+(c_k-d_k)}\right\rfloor, k=0,1,\ldots,6.$

Proof:

First, we introduce an impulse train modulated by the samples $f(nT)$ of the signal $f(t)$:

$$f_\delta(t)=\sum_{n=-\infty}^{\infty} Tf(nT)\delta(t-nT).$$

According to the Poisson formula, the Fourier transform of $f_\delta(t)$ is given by

$$F_\delta(\omega)=\sum_{n=-\infty}^{\infty} Tf(nT)e^{-jnT\omega}=\sum_{n=-\infty}^{\infty} F(\omega+n\omega_s) \tag{4-5}$$

where $F(\omega)$ is the Fourier transform of $f(t)$. The spectrum of $f(t)$ can be recovered from (4-5) by applying the ideal two-band bandpass filter (4-2) to $F_\delta(\omega)$ as follows:

$$F(\omega)=S(\omega)F_\delta(\omega)=S(\omega)\sum_{n=-\infty}^{\infty} F(\omega+n\omega_s). \tag{4-6}$$

This is guaranteed because none of the spectra $F(\omega+n\omega_s)$, $n\neq 0$, overlap with $F(\omega)$ at the sampling rate ω_s satisfying (4-4) according to Theorem 3.1 and earlier discussions. Therefore, taking the inverse Fourier transform of (4-6) yields

$$f(t)=s(t)*f_{\delta}(t)=\sum_{n=-\infty}^{\infty} f(nT)\left\{\frac{\sigma_1 T}{\pi}\text{csinc}_{[\omega_1,\sigma_1]}(t-NT)+\frac{\sigma_2 T}{\pi}\text{csinc}_{[\omega_2,\sigma_2]}(t-NT)\right\}$$

$$=\sum_{n=-\infty}^{\infty} f(nT)\left\{\frac{2\sigma_1}{\omega_s}\text{csinc}_{[\omega_1,\sigma_1]}(t-NT)+\frac{2\sigma_2}{\omega_s}\text{csinc}_{[\omega_2,\sigma_2]}(t-NT)\right\}$$

■

To give the reader an example of the difference in sampling rates, consider a signal bandpassed to $[20kHz, 25kHz] \cup [500kHz, 520kHz]$. Then the Nyquist rate or industry standard is $1040kHz$. However, with the proposed sampling mechanism, the sampling rate can be as low as $50kHz$. This sampling rate was determined by the MatLab software appended. The difference is significant. The sampling rate has thus been reduced by a factor of twenty. When considering higher frequency signals i.e. mHz, the sampling rate can be reduced to kHz. This reduced rate is commercially appealing when considering expensive sampling devices, see to Fig 1.2.

CHAPTER 5

CONCLUSION AND FUTURE WORK

In this thesis, we considered all possible step sizes for the sampling frequency of two-band passband signals. In chapter 2, we partitioned the positive real line into subintervals and determined all the feasible stepsizes from the subintervals. The propositions led to the necessary and sufficient conditions for the permissible stepsize for the shifting without intersection of two arbitrary disjoint intervals. The closed form expressions for the permissible stepsizes from Theorem 2.1 along with the fast algorithm developed in chapter 3 allow us to calculate the optimal and feasible sampling rates for the two-band passband signals, which consists of six pairs of sub-bands. Finally, an example in chapter 4 was used to demonstrate the efficacy of the new sampling scheme.

The new sampling scheme will reduce the sampling rate dramatically in most cases. This work can be extended to multi-band passband signals and the computation of the optimal and feasible solutions will be low-complexity allowing for implementation in real design. The simplicity of first-order sampling makes this approach very attractive due to the minimal cost for designing the sampling device. The range of permissible sampling rates calculated by this scheme allows for flexibility in design without using guard-band. Overall, this new approach will be viable and commercially appealing.

REFERENCES

[1] A. J. Coulson, R. G. Vaughan, M. A. Poletti, Frequency-shifting using bandpass sampling, *IEEE Trans. Signal Processing*, v. 42, 1556-1559, 1994.

[2] J. D. Gaskell, *Linear Systems, Fourier Transforms, and Optics*, Wiley, New York, 1978.

[3] W. D. Gregg, *Analog and Digital Communications Systems*, Wiley, New York, 1978.

[4] T. K. Moon, Exact reconstruction of samples of signal from samples of its integral, *Electronics Letters*, v. 36, no. 12, 1079-1081, 2000.

[5] S. K. Mitra, A. Mahalanobis, and T. Saramaki, A Generalized Structural Subband Decomposition of FIR Filters and Its Application in Efficient FIR Filter Design and Implementation, *IEEE Trans. Circuits and Systems*, v. 40, n. 6, 1993.

[6] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing, Principles, Algorithms, and Applications*, Prentice Hall, New Jersey, 1996.

[7] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*, California Technical Publishing, San Diego, California, 1997.

[8] R. G. Vaughan, N. L. Scott, and D. R. White, The Theory of Bandpass Sampling, *IEEE Trans. Signal Processing*, v. 39, 1973-1984, 1991.

[9] Y. Wu, A Universal Interpolative Filter for Lowpass and Bandpass Signals-CSINC Interpolator, *J. of Digital Signal Processing*, v. 15, 425-436, 2005.

[10] C. Xiao, Reconstruction of Bandlimited Signal with Lost Samples at Its Nyquist Rate-The Solution to a Nonuniform Sampling Problem, *IEEE Trans. Signal Processing*, v. 43, no. 4,1008-1009,1995.

APPENDIX

MATLAB CODE FOR TWO CALCULATING FEASIBLE STEP SIZE INTERSECTIONS

```
%%This method allows user to input band positions and calls feasibleSol
function deltaCalculator = calculateDelta(a,b,c,d)
t1 = floor((a/(b-a)));
j=1;
k=1;
l=1;
m=1;
for i=0:(t1-1)
    x1(j) = 2*b/(t1 + 1 - i);
    x1(j+1) = 2*a/(t1 - i);
    j=j+2;
end
    x1(2*t1+1) = 2*b;
    x1(2*t1+2) = 9999999999;
t2 = floor(((a+c)/((b-a)+(d-c))));
for i=0:(t2-1)
    x2(k) = (b+d)/(t2 + 1 -i);
    x2(k+1) = (a+c)/(t2 - i);
    k=k+2;
end
    x2(2*t2+1) = b+d;
    x2(2*t2 + 2) = 9999999999;
t3 = floor(((c-b)/((b-a)+(d-c))));
for i=0:(t3-1)
    x3(l) = (d-a)/(t3 + 1 - i);
    x3(l+1) = (c-b)/(t3 -i);
    l=l+2;
end
    x3(2*t3+1) = d-a;
    x3(2*t3+2) = 999999999;
t4 = floor((c/(d-c)));
for i=0:(t4-1)
    x4(m) = 2*d/(t4 + 1 - i);
    x4(m+1) = 2*c/(t4 - i);
    m=m+2;
end
    x4(2*t4+1) = 2*d;
    x4(2*t4+2) = 999999999;

deltaCalculator = feasibleSol(x1,x2,x3,x4);


%%This method takes vectors x1,x2,x3,x4 consisting of intervals and
determines intersection.
function feasible = feasibleSol(x1,x2,x3,x4)
i=1;
j=1;
k=1;
n=1;
p=1;
```

```
while                (i<=length(x1)-1)&(j<=length(x2)-1)&(k<=length(x3)-1)&(n<=length(x4)-1)
   maxim = maximum(x1(i),x2(j),x3(k),x4(n));
   minum = minimum(x1(i+1),x2(j+1),x3(k+1),x4(n+1));
  if minum >= maxim
   if x1(i) == maxim
    if x1(i+1) == minum y(p)=x1(i);y(p+1)=x1(i+1);i=i+2;p=p+2;
       elseif x2(j+1) == minum y(p)=x1(i);y(p+1)=x2(j+1);j=j+2;p=p+2;
       elseif x3(k+1)==minum y(p)=x1(i);y(p+1)=x3(k+1);k=k+2;p=p+2;
              else  y(p)=x1(i);y(p+1)=x4(n+1);n=n+2;p=p+2;
              end

       elseif x2(j) == maxim
          if x2(j+1) == minum y(p)=x2(j);y(p+1)=x2(j+1);j=j+2;p=p+2;
              elseif            x1(i+1)            ==            minum y(p)=x2(j);y(p+1)=x1(i+1);i=i+2;p=p+2;
                    elseif          x3(k+1)           ==           minum y(p)=x2(j);y(p+1)=x3(k+1);k=k+2;p=p+2;
                       else y(p)=x2(j);y(p+1)=x4(n+1);n=n+2;p=p+2;
                         end

        elseif x3(k) == maxim
          if x3(k+1) == minum y(p)=x3(k);y(p+1)=x3(k+1);k=k+2;p=p+2;
          elseif            x2(j+1)            ==             minum y(p)=x3(k);y(p+1)=x2(j+1);j=j+2;p=p+2;
          elseif            x1(i+1)            ==             minum y(p)=x3(k);y(p+1)=x1(i+1);i=i+2;p=p+2;
          elseif            x4(n+1)            ==             minum y(p)=x3(k);y(p+1)=x4(n+1);n=n+2;p=p+2;
                     end

         elseif x4(n) == maxim
               if            x4(n+1)            ==            minum y(p)=x4(n);y(p+1)=x4(n+1);n=n+2;p=p+2;
               elseif          x2(j+1)            ==            minum y(p)=x4(n);y(p+1)=x2(j+1);j=j+2;p=p+2;
               elseif          x3(k+1)            ==            minum y(p)=x4(n);y(p+1)=x3(k+1);k=k+2;p=p+2;
               elseif          x1(i+1)            ==            minum y(p)=x4(n);y(p+1)=x1(i+1);i=i+2;p=p+2;
               end
   end

  else
      if x1(i+1)==minum i=i+2;
      elseif x2(j+1)==minum j=j+2;
      elseif x3(k+1)==minum k=k+2;
      else n=n+2;
      end
end
end
   feasible = y(1);
```

JAVA SIMULATION FOR FEASIBLE STEP SIZES

%%This new set of code is a java simulation of spectra copies on the frequency axis

```
import java.awt.*;

import javax.swing.*;

public class SimulationFrame extends JFrame{

    double a;

    double b;

    double c;

    double d;

    gridPanel grid;

    public SimulationFrame(double a, double b, double c, double d){

    this.a = a; this.b = b; this.c = c; this.d = d;

    grid = new gridPanel(a,b,c,d);

    getContentPane().add(grid);

     }

    public void paintComponent(Graphics g){

                super.paint(g);

                g.drawLine(100,100,25,50);

                g.setColor(Color.black);

                      repaint();

              }}

import java.awt.*;

import javax.swing.*;
```

```
public class Simulation{
    public static void main(String[] args){
    String a = JOptionPane.showInputDialog(null,"Input a");
    String b = JOptionPane.showInputDialog(null,"Input b");
    String c = JOptionPane.showInputDialog(null,"Input c");
    String d = JOptionPane.showInputDialog(null,"Input d");
   double a1 = Double.parseDouble(a);
    double b1 = Double.parseDouble(b);
    double c1 = Double.parseDouble(c);
    double d1 = Double.parseDouble(d);
    double a2 = (int)(a1*100);
    double b2 = (int)(b1*100);
    double c2 = (int)(c1*100);
    double d2 = (int)(d1*100);
    if(a2%10<=4)  {a2 = ((int)(a2/10));a2 = a2/10;}
    else {a2 = ((int)(a2/10));a2 = a2/10 + .05;}
    if(b2%10<=4)  {b2 = ((int)(b2/10));b2 = b2/10;}
    else {b2 = ((int)(b2/10));b2 = b2/10 + .05;}
    if(c2%10<=4)  {c2 = ((int)(c2/10));c2 = c2/10;}
    else {c2 = ((int)(c2/10));c2 = c2/10 + .05;}
    if(d2%10<=4)  {d2 = ((int)(d2/10));d2 = d2/10;}
    else {d2 = ((int)(d2/10));d2 = d2/10 + .05;}
    SimulationFrame fram1 = new SimulationFrame(a2,b2,c2,d2);
```

```
  fram1.repaint();
  fram1.setSize(1100,1100);
  fram1.setVisible(true)}}
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class gridPanel extends JPanel implements ActionListener{
   int delta;
   box Box1;
   box Box2;
   JButton bt1;
        JButton bt2;
   JButton bt3;
   JButton bt4;
   JTextField field;
   int t;
   double array[];
   double a;
   double b;
   double c;
   double d;
   int x1;
   int y1;
```

```
int x2;
int y2;
Timer timer;
int i;
public gridPanel(double a, double b, double c, double d){
  Box1 = new box(a,b,Color.red);
  Box2 = new box(d,c,Color.red);
  timer = new Timer(1000,this);
  bt1 = new JButton("Start");
  bt2 = new JButton("Guess");
  bt3 = new JButton("Stop");
  bt4 = new JButton("Restart");
  bt1.setMnemonic('s');
  bt3.setMnemonic('p');
  bt4.setMnemonic('r');
  this.t = (int)(Math.floor((a-c)/((b-a)+(c-d))));
  array = new double[2*t];
  for(int i=0;i<2*t;i+=2){
     array[i] = (b-d)/(t+1-(i/2));
     array[i+1] = (a-c)/(t-(i/2));
        }
  bt1.addActionListener(this);
  bt2.addActionListener(this);
```

```
 bt3.addActionListener(this);
 bt4.addActionListener(this);
 field = new JTextField();
this.a = a; this.b = b; this.c = c; this.d = d;
 add(bt1);
        add(bt2);
                add(bt3);
 add(bt4);
 add(field);
 i = 0;
 x1 = (int)(20*a+500);
 y1 = 80;
 x2 =  (int)(20*d+500);
 y2 = 80;
 double delta1 = (b-d)/(Math.floor((a-c)/((b-a)+(c-d)))+1);
 this.delta = (int)(20*delta1) + 1;
 this.requestFocus();
}
public void setBox(box Box,int x, int y){
 Box.setLocation(20,30);
}
public void paintComponent(Graphics g){
super.paintComponent(g);
```

```
g.drawLine(0,100,1000,100);
g.setColor(Color.black);
for(int i=0;i<=50;i++){
          g.drawLine(20*i,105,20*i  ,95);
String n = Integer.toString(i-25);
  g.drawString(n,20*i,120);
  g.setColor(Color.black);
         }
   g.drawString("Enter a Guess For Delta",430,270);
   field.setLocation(450,300);
   field.setSize(100,20);
   add(Box1);
   Box1.setLocation((int)(20*a + 500),y1);
   add(Box2);Box2.setLocation((int)(20*d + 500),y2);
   Box1.setSize((int)(20*(b-a)),20);
   Box2.setSize((int)(20*(c-d)),20);
   g.setColor(Color.blue);
   g.fillRect(x1,y1 -20,(int)(20*(b-a)),20);
   g.setColor(Color.blue);
   g.fillRect(x2,y2 - 20,(int)(20*(c-d)),20);
   g.setColor(Color.black);
   g.drawString("Permissable Step Size",400,350);
   for(int i=0;i<2*t;i+=2){
```

```
g.drawString("[" + Double.toString(array[i]),550,350 + (i/2)*20);
g.drawString(Double.toString(array[i+1])+"]",580,350 + (i/2)*20);
g.drawString("["+Double.toString(b-d),550,350 + t*20);
 g.drawString("Inf]",580,350 + t*20);
}}
public void actionPerformed(ActionEvent e){
  if(e.getSource()==bt1)
   double del = (b-d)/(Math.floor((a-c)/((b-a)+(c-d)))+1);
  delta = (int)(20*del) + 1;
          timer.start();
              x1 = x1 + delta;
              x2 = x2 + delta;
    repaint();
  }
  if(e.getSource()==bt2){String text = field.getText();
     double del = Double.parseDouble(text);this.delta = (int)(20*del)+1;
     timer.start();
     }
  if(e.getSource()==bt3){timer.stop();
  }
  if(e.getSource()==timer){
                    i = i + 1;
                    x1 = x1 + delta;
```

```
                                        x2 = x2 + delta;
                repaint();  }
            if(e.getSource()==bt4){
            x1 = (int)(20*a+500);
            y1 = 80;
            x2 =  (int)(20*d+500);
            y2 = 80;
            repaint();  }}}
import javax.swing.*;
import java.awt.*;
class box extends JPanel{
   double a;
   double b;
   double x;
   double y;
   Color c;
   public box(double a, double b, Color c){
                  this.a = a;
                  this.b = b;
                  this.c = c;
                   setSize((int)(20*(b-a)),20);
                   }
   public void moveBox(int x, int y){
```

```
    setLocation(x,y);}
public void paintComponent(Graphics g){
            super.paintComponent(g);
            g.setColor(Color.black);
            g.fillRect(0,0,(int)(20*(b-a)),20);   }}
```