On Methods for the Suppression of Aliasing in Time-Series Gene Expression Data

by

Alex Mckenzie, B.Eng.

A thesis submitted to the
Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of

Master of Applied Science in Biomedical Engineering

Ottawa-Carleton Institute for Biomedical Engineering
Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario
September, 2010

Your file Votre référence
ISBN: 978-0-494-71517-8
Our file Notre référence
ISBN: 978-0-494-71517-8

Abstract

Gene regulatory networks are an important topic in molecular biology, and their discovery is an area of great activity. Time Series Gene Expression Studies (TSGES), in which multiple sequential measurements of Messenger RNA (mRNA) levels are made to elucidate these networks, are complicated by the extremely high cost of mRNA measurements.

The high cost of mRNA measurements necessarily limits the number of samples that can be measured in a single experiment. For an experiment of fixed duration, a limited number of measurements implies a minimum sampling interval, which has the potential to violate the Nyquist criterion. In this thesis a simulation of the GAL regulon in *Saccharomyces cerevisiae* (*S. cerevisiae*) was used to demonstrate that signal corruption from aliasing of high-frequency signal components is possible.

Effective signal analysis under conditions of restricted sensing is a well-studied area in digital signal processing, and the lessons learned there can be usefully applied to biological research. In particular, the techniques of jitter sampling and Time Aggregation and Skip Sampling (TASS) have great potential to reduce data distortion due to aliasing.

In jitter sampling, the actual time each measurement is taken is deviated by

a small, random amount from the nominal periodic sampling time. Since the probability of the contribution of an above-Nyquist criterion frequency exactly matching all of a number of small, stochastic deviations is vanishingly small, signal contributions which would otherwise be aliased are suppressed. This technique was tested against both a simple sinusoidal data model and the above-mentionned GAL regulon simulation. These tests confirmed that jitter sampling could remove in excess of half of the aliased content of a signal: in one case, an aliased signal was reduced to 25% of the intensity of a non-aliased signal originally of equal magnitude. Furthermore, jitter sampling was demonstrated to be effective with as few as ten samples and when applied to biological systems.

TASS, in which a cluster of multiple samples are taken in temporal proximity to each primary measurement timepoint and averaged together, is also applicable. Rapid variations in the underlying signal are "averaged out", suppressing high frequency components of the signal. While it is very expensive to measure mRNA levels, the collection of additional physical samples is nearly free. In a biological context, therefore, averaging can be done at very low cost by physically mixing samples together before measuring mRNA levels. Like jitter sampling, TASS was tested on both simplistic and realistic simulated data. One implementation of TASS was able to reduce the aliased signal intensity by 98% relative to the unaliased signal. This technique was also effective with few samples and in biological systems.

TASS and jitter sampling, while accomplishing similiar objectives, have marked differences and the choice between them is non-trivial. Jitter sampling is simpler to implement and more selective in effect than TASS, but it is counterintuitive and can be unpredictable. Implementing TASS can be more complex and costly, and a poor choice of parameters can result in either inadequate or excessive suppression; on the

other hand, TASS is predictable and easy to understand.

To my family.

Acknowledgments

The author gratefully acknowledges his supervisors, Professors Jim Green and Richard Dansereau, for their their valuable insight, advice and direction throughout his graduate studies.

He also would like to thank Mads Kaern and Vida Abedi, of the Ottawa Institute for Systems Biology, who suggested and refined the GAL regulon model used to validate this work.

This research was partially funded by an Ontario Graduate Scholarship.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

TASS	Time Aggregation and Skip Sampling
TSGES	Time Series Gene Expression Studies
GRN	Gene Regulatory Network
ODE	Ordinary Differential Equation
FT	Fourier Transform
FFT	Fast Fourier Transform
L-S	Lomb-Scargle
LSSA	Least-Squares Spectral Analysis
DNA	Deoxyribonucleic Acid
RNA	Ribonucleic Acid
mRNA	Messenger Ribonucleic Acid (RNA)
RNAi	RNA Interference
siRNA	Small Interfering RNA
RISC	RNA-induced Silencing Complex
DSP	Digital Signal Processing

LPF	Low-Pass Filter
S. cerevisiae	*Saccharomyces cerevisiae*
E. coli	*Escherichia coli*
TZD	Thiazolidinedione
PPAR-γ	Peroxisome Proliferator-Activated Receptor Gamma
cAMP	Cyclic Adenosine Monophosphate
CRP	Cyclic Adenosine Monophosphate (cAMP) Receptor Protein

Chapter 1

Introduction

An organism's genome, made of Deoxyribonucleic Acid (DNA) and arranged into genes, is a complete blueprint of all proteins used by the organism. This blueprint is converted from DNA into proteins through a process known as expression, though not all genes are expressed equally. To determine which genes are expressed when, and the relationships between them, a form of experiment known as a Time Series Gene Expression Studies (TSGES) can be conducted. In this type of experiment, a device known as a microarray is used to measure the expression levels of many genes simultaneously. A sequence of such measurements is made over time, using one microarray (which are not re-useable) for each time point.

1.1 Motivation

Due to the high cost of Messenger RNA (mRNA) microarrays, TSGES are necessarily limited in the number of time points they can measure. As a result, presuming a fixed budget, investigators must play a balancing act between the temporal scope and temporal resolution of their study. This implies the possibility of violating the Nyquist criterion.

The field of Digital Signal Processing (DSP) has long been concerned with the

problem of effective signal analysis under conditions of restricted sensing. Two DSP techniques, jitter sampling and Time Aggregation and Skip Sampling (TASS), have great potential to reduce data distortion due to aliasing.

Jitter sampling works by varying the actual time each measurement is taken by a small amount. Since the probability of the contribution of an above-Nyquist criterion frequency exactly matching all of a number of small, stochastic deviations is vanishingly small, signal contributions which would otherwise be aliased are suppressed.

Collecting biological samples is generally inexpensive; the measurement of mRNA levels in those samples is the costly component of TSGESs. TASS takes advantage of this by collecting multiple samples in temporal proximity to each primary measurement timepoint and averaging these together. High frequency components of the signal are supressed because rapid variations in the underlying signal are "averaged out". In a biological context, averaging can be done at very low cost by physically mixing samples together before measuring mRNA levels.

1.2 Problem Statement

This thesis aims to develop advanced DSP techniques for reducing aliasing in TSGES without increasing the number of microarrays, and thus the cost, involved.

1.3 Contributions

This work makes three major contributions:

- It establishes that aliasing can occur in TSGES conducted using typical sampling rates from published studies.
- It shows that the appropriate application of jitter sampling can reduce aliasing in TSGES.

- It demonstrates that TASS is a cost-effective technique capable of mitigating aliasing in TSGES.

1.4 Thesis Organization

The following four chapters lay a foundation for and present the contributions described above. In Chapter 2, foundational theoretical material from biology and DSP is discussed and relevant prior work from the literature is reviewed. An illustration of aliasing in biological systems, using a simulation of the GAL regulon, is presented in Chapter 3. The two techniques advocated in this work, jitter sampling and TASS, are presented in Chapter 4 and Chapter 5, respectively. These techniques are compared in Chapter 6. Chapter 7 summarizes the results and possibilities for future research.

Chapter 2

Background

This chapter discusses background material which helps put this research in context. Section 2.1 provides an overview of the relevant elements of cell biology while Section 2.2 discusses certain useful signal processing concepts. Finally, Section 2.3 describes other published research focused on the problem of optimizing the use of microarrays in TSGES.

2.1 Biology

2.1.1 Basic Genetics

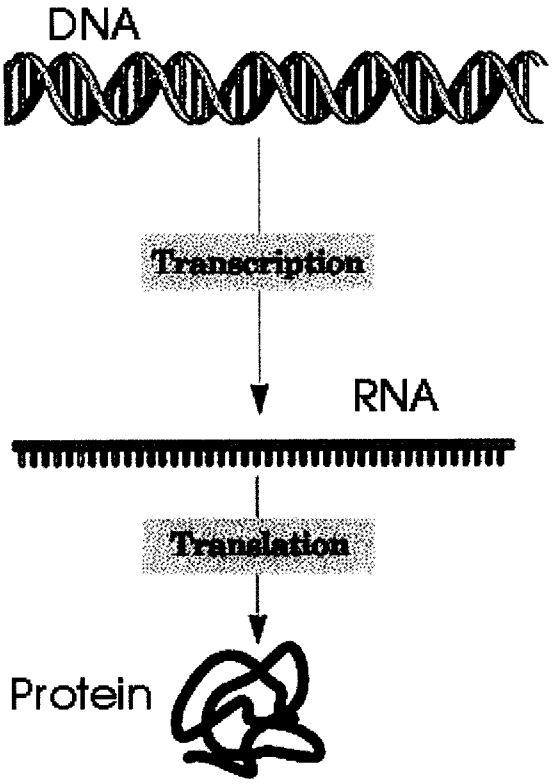


Figure 2.1: Information flow in eukaryotes.
Image credit: [1]

Information flow in an organism follows a specific path, as illustrated in Figure 2.1. The process by which stored genetic "blueprints" are converted into useful proteins is known as gene expression; it consists of two main steps, transcription and translation.

The permanent repository of information in nearly all extant organisms is nuclear DNA. Information is stored in the form of a long sequence of base-pairs inside the cell nucleus. DNA consists of a pair of phosphate-sugar backbones, arranged in the now-famous double helix. Attached to these backbones are complementary pairs of nucleotides. The four nucleotides present in DNA are adenine (A), thymine (T),

guanine (G) and cytosine (C). A pairs with T, and C with G.

In a process called transcription, this DNA-encoded information is copied into complementary mRNA. RNA uses the same nucleotides as DNA, except that thymine is replaced with uracil. mRNA is produced on demand, one gene at a time.[1] Each mRNA strand, therefore, codes for a single protein which the cell presumably needs at that time. The mRNA moves from the nucleus to the cytoplasm. Once in the cytoplasm, a ribosome attaches to the mRNA, and the information in the mRNA is translated into protein. After translation is complete, the mRNA strand (which is not consumed in this process) detaches from the ribosome, at which point it can again be processed by a ribosome. The mRNA strand will persist in the cytoplasm and be used to generate additional copies of its corresponding protein until it is broken down by enzymes present in the cell.

While this simplistic description of the process of gene expression would tend to lead one to believe that mRNA levels and protein levels are highly correlated, this is not always the case [2, 3, 4]. In fact, mRNA levels typically are only 40% ($r = 0.6$) predictive of protein levels, and for some genes, correlation can actually be negative [2]. There have been some efforts to understand what factors affect this relationship [5], but as yet, this is a poorly understood area. Notwithstanding these issues, mRNA concentrations remain a useful proxy for gene expression levels, mostly due to the ease with which the mRNA levels for large numbers of different genes can be measured in parallel (see Section 2.1.3).

2.1.2 Gene Regulatory Networks and their Discovery

Gene Regulatory Networks (GRNs) consist of an interrelated set of genes, in which the activity of one or more of the genes is up- or down-regulated by others in the set. GRNs commonly serve as "decision switches", turning synthesis of particular proteins

[1] A gene is defined as a DNA sequence encoding a single protein or a defined set of related proteins.

on or off in response to changing environment conditions (see, for example, the *lac* operon in Section 2.1.5 and the GAL regulon in Chapter 3). GRNs are also involved in internal cell functions, including regulation of the cell cycle and tumor suppression (see [6, 7, 8, 9, 10] and Section 2.1.6).

As discussed in Section 2.1.1, mRNA is transcribed per gene and on demand. As a result, the concentration of mRNA transcripts of a particular gene can give an approximate indication of the demand for the protein encoded by that gene[2]. Repeated measurements of the mRNA transcript concentrations of a large set of different genes (see Section 2.1.3) in an organism over an extended period of time can reveal patterns and connections between the expression of different genes. Frequently, these patterns are just simple correlations, but can be more complex, depending on the particular algorithm used. These patterns are believed to represent GRN links.

A number of different computational techniques have been used to find such patterns (this search is known as GRN reverse engineering). A review of several of the most prominent techniques (relevance networks, graphical Gaussian models and Bayesian networks) and a direct, head-to-head trial (rare in the literature) can be found in [11]. Also worthy of note is [12], describing a software application which implements a number of reverse engineering techniques in a single, comparatively straightforward tool. The technique proposed in [13], while not dramatically superior to competing alternatives, is interesting in that it applies DSP techniques to this problem: the gene expression levels are treated as discrete time-invariant signals and compared with a view to finding signals which appear to be phase-shifted versions of each other.

[2]But see the last paragraph of Section 2.1.1.

Meta-Genes

TSGES typically involve tens of timepoint measures of thousands of genes. As a result, any analysis is severely underdetermined. A popular technique for reducing the dimensionality is to form data clusters of individual genes as "metagenes". Clustering algorithms are used to reduce thousands of genes to a set of perhaps a few dozen metagenes with similar expression profiles over time.

A large number of competing clustering algorithms have been proposed, for example: [14] focuses on clustering when the data are unevenly sampled, [15] presents a method for hierarchical clustering with guaranteed optimal leaf ordering and [16] (by the same group) refines the method by allowing the number siblings to be specified. Also see [17], which proposes a framework for evaluating the validity of clustering algorithms.

2.1.3 Time-Series Gene Expression Studies

TSGES are an integral part of the discovery of GRNs; they are used to obtain the raw data needed for the algorithms described above. In a TSGES, mRNA levels are measured repeatedly at sequential time points using microarrays.

Microarrays

Microarrays are the standard tool used to measure mRNA levels in a TSGES. The essence of a microarray consists of a set of short DNA strands (called probes) bonded to a substrate (typically glass or silicon). The probes are complementary to the target (mRNA sequence of interest), and are typically laid out in a grid pattern, with each small region (called a "spot") containing probes for a different target.

Procedure

At each time point, a sample[3] is taken from the subject of the study. The form of the sample varies according to the nature of the study and the organism; for example, studies in humans typically involve simple blood samples, while studies of unicellular organisms will typically use a portion of a homogeneous culture.

These samples are immediately biologically "frozen" using a buffer solution which inhibits biological activity. The cells are then broken down and the mRNA extracted. The mRNA is purified, (optionally) amplified and tagged with a fluorescent marker. The sample is then applied to a microarray, and "washed off". Any mRNA which is complementary to the microarray probes will "stick" to the array, whereas non-matching strands will be removed.

The array is scanned while lasers excite the fluorescent markers. The intensity of the fluorescence at each spot is proportional to the quantity of corresponding mRNA present in the sample. The image is digitally processed to remove noise from the image, quantify the intensity of each spot and normalize the data.

2.1.4 Regulation Mechanisms

Gene regulation can occur at any stage of the gene expression process described in Section 2.1.1. While this area is not yet well understood, a handful of specific mechanisms are at least moderately well understood. Note that this section does not presume to discuss all regulatory mechanisms. Furthermore, it should be noted that most attempts at reverse engineering or simulating GRNs ignore the subtlety of varied regulation mechanisms, and simply describe associations, without attempting to identify specific mechanisms. While not accounted for in most models, the differences between the different regulation mechanisms is likely to affect the behaviour of the

[3]Typically, multiple duplicate samples will be taken for cross-validation.

GRN. Different mechanisms operate at widely varying rates (from seconds to days), and may have more complex kinetics than are currently understood.

Transcription Factors

Gene expression can be regulated by varying the rate of transcription. This is governed by proteins known as transcription factors. These proteins, classified as either activators or repressors, bind to specific target sites on the DNA strand upstream of the target gene. Once bound to its target site, a transcription factor will modify the activity of RNA polymerase in the region of the target gene. Activators act to increase gene transcription by increasing the binding affinity or activity of RNA polymerase on the target gene, whereas repressors block or inhibit RNA polymerase and thereby reduce gene transcription.

RNA Interference

Translation can be blocked by Small Interfering RNA (siRNA) in a process known as RNA Interference (RNAi). siRNA are short ($\leq$ 25 base) double-stranded RNA segments which are complementary to the mRNA sequence they inhibit.

In the best understood case, the siRNA fragment combines with a catalyst protein called argonaute to form an RNA-induced Silencing Complex (RISC). The guide strand (the strand which is complementary to the target) portion of the RISC then binds to the matching section of its target mRNA strand, preventing translation of the strand.

RNA degradation

The persistence of an mRNA strand determines, in large part, how many copies of its corresponding protein are produced. Enzymes present in cytoplasm to break down mRNA, and their activity is modulated by various factors. One end (the 5' end) of

the mRNA strand is "capped", which prevents it from being degraded. The other end has a long string of adenosine bases (known as a polyA tail) added to it, as a "sacrificial" protective buffer. Various sequences within the strand can also play a role, by either speeding or delaying degradation, though the mechanisms involved are not well understood.

Epigenetics

Epigenetics refers to heritable traits governed by mechanisms other than the DNA sequence. Typically, this means heritable forms of gene regulation. The best known such mechanism is methylation, which inactivates a given sequence of DNA by semi-permanently binding a methyl group to it. Epigenetic regulation mechanisms tend to be long-lasting and persistent, rarely changing in the comparatively short time-scale of TSGES. They are, therefore, outside the scope of this document.

2.1.5 An Example Gene Regulatory Network: the *lac* operon

The concept of a GRN is perhaps best understood with an example. The *lac* operon, being probably the first known GRN, is an extremely well known and studied GRN, appearing in virtually every university-level introductory biology textbook. Reviewing this network will also facilitate understanding of the conceptually similar, but much more complex GAL regulon, which will be introduced in Chapter 3 and figure prominently in this work.

The *lac* operon GRN controls the metabolism of sugars in *Escherichia coli* (*E. coli*). Specifically, it regulates the production of enzymes necessary to the metabolism of the sugar lactose. The production of these enzymes incures a metabolic cost, and so it is advantageous for *E. coli* to do so only when necessary. Specifically, these enzymes are only produced when lactose is present and a preferred energy source (glucose) is not. Figure 2.2 summarizes this behaviour.

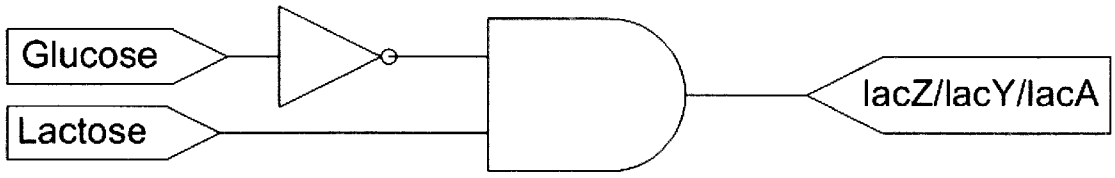


Figure 2.2: A conceptual representation of the *lac* operon as a digital logic system.

The *lac* operon includes five important components:

- a **promoter** *(A region on the DNA strand to which another species can bind, and thereby increase transcription.)*
- an **operator** *(A region on the DNA strand to which another species can bind, and thereby inhibit transcription.)*
- three structural genes: **lacZ**, **lacY** and **lacA**

Additionally, several other species[4] are important to the behaviour of the *lac* operon:

- cAMP Receptor Protein (CRP)
- a **repressor** protein
- **allolactose**

These components co-operate, as shown in Figure 2.3 and described below, to regulate the metabolism of lactose.

[4]In a biochemistry context, *species* refers generically to the various substances (proteins, nucleic acids, lipids, sugars, complexes, etc.) under consideration. This is unrelated to its usage in taxonomy.

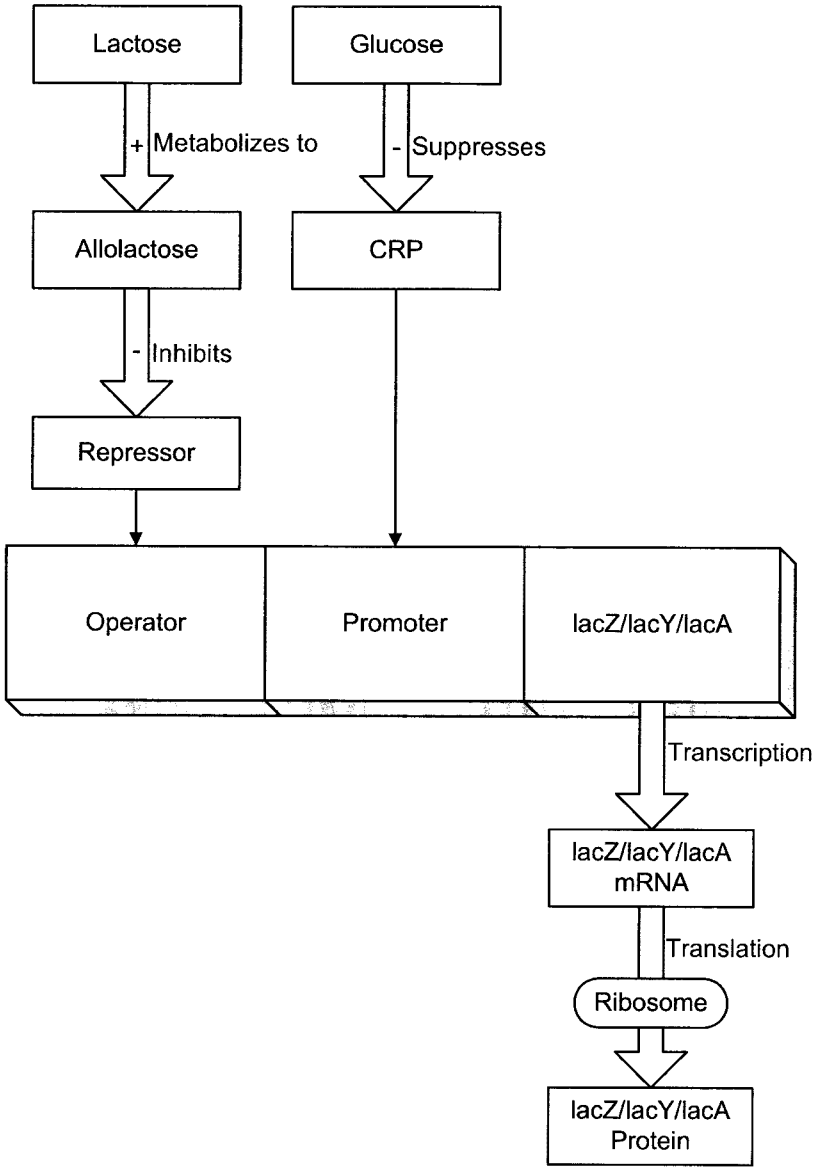


Figure 2.3: Biochemical pathways in the *lac* operon.

When lactose is absent, the repressor is active and binds to the operator. This inhibits transcription of the structural genes, and very little of the lactose metabolism enzymes are produced.

Allolactose, a metabolite of lactose, is present when lactose is available to the organism as a food source. Allotlactose binds to a receptor site on the repressor protein, inactivating it. This permits increased transcription of the structural genes,

though enzyme production levels also vary according to the availability of glucose. If glucose is absent, significant amounts of CRP will be produced[5] which will bind to the promoter region of the *lac* operon, resulting in the production of large amounts of the lactose metabolism enzymes. If glucose is present, little or no CRP will be available. Without the promoter being active, little of the lactose enzymes will be produced. These different cases are illustrated in Figure 2.4.

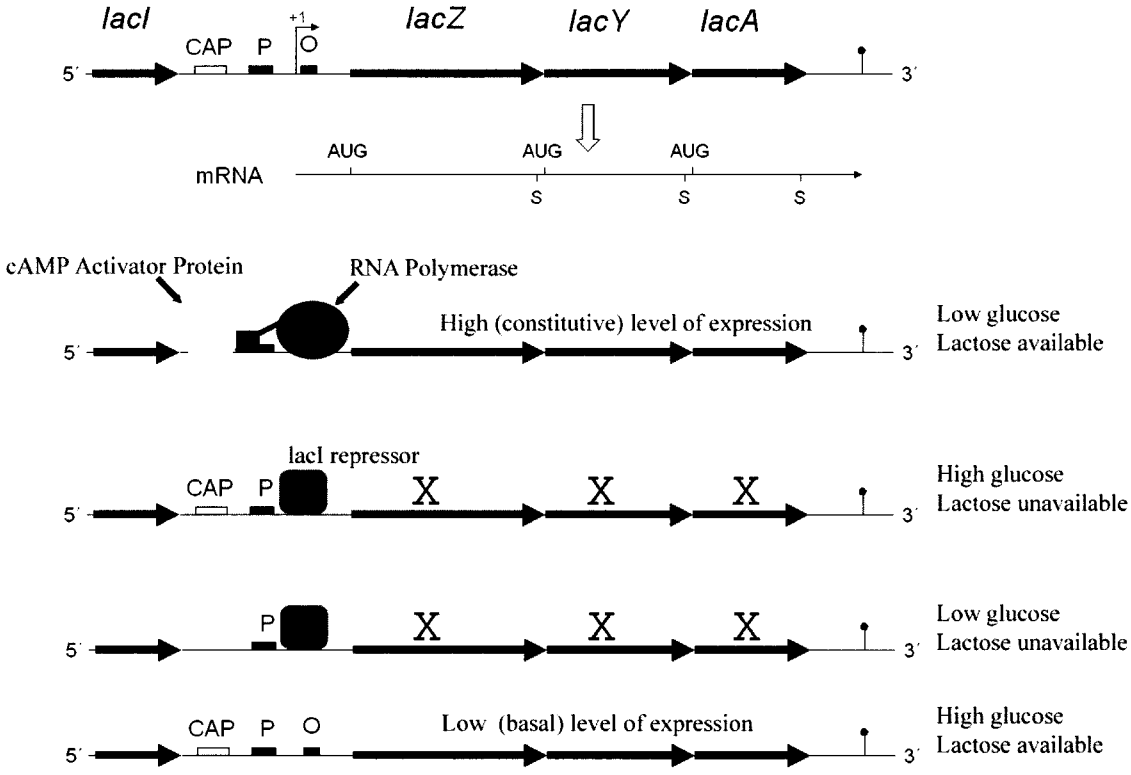


Figure 2.4: Behaviour of the *lac* operon.
Image credit: [18]

[5]The biochemical pathway between glucose and CRP is beyond the scope of this discussion: it is sufficient that the concentration of CRP is inversely related to, and governed by, the concentration of glucose.

2.1.6 The Significance of GRNs

The discovery and analysis of GRNs is a topic of much interest in biology and medicine, and with good reason. GRNs are believed to be clinically significant in many important areas of human health. Many drugs are thought to work by modulating one or more regulatory pathways in a GRN. Similarly, the malfunction of one or more GRNs is believed to be necessary for the formation of a cancerous tumour [6].

A class of drugs known as Thiazolidinediones (TZDs) increase insulin sensitivity and are a standard treatment for type-II Diabetes Mellitus [19]. All drugs in this class modify gene transcription by binding to a nuclear receptor known as Peroxisome Proliferator-Activated Receptor Gamma (PPAR-γ). While this basic mechanism is well understood, there are significant complexities that still elude researchers. *In vitro* studies, such as [20], have shown that different drugs in this class have notably different effects on gene expression levels in human cells. These differences are not just a matter of academic curiosity: a direct comparison [21] of patients taking either of two such drugs (rosiglitazone and pioglitazone) showed that there was a 15% difference in mortality between the two groups, despite the fact that these drugs, in theory, operate by exactly the same mechanism. This is a case where it is clear that GRN effects have a direct relevance to patient survival.

2.2 Digital Signal Processing

2.2.1 Lomb-Scargle

The Lomb-Scargle (L-S) method is used for analyzing the frequency content of a signal. In this respect, it is similar to the Fourier Transform (FT) (and, by extension, the Fast Fourier Transform (FFT)). The most important advantage of L-S over the

FT is that it can be used to analyze unevenly sampled signals.

L-S is a variation of Least-Squares Spectral Analysis (LSSA).[6] These techniques work by comparing the fit of generated sinusoids to the actual signal, and choosing the best-fitting set.

The use of the L-S periodogram in this work relied on [22], a publically available implementation. Initially, this periodogram was adopted in preference to the FFT due to the belief that it was necessary to account for the deviations of jitter sampling (see Chapter 4) when determining the measured frequency content of the signal. While later testing showed negligible benefit (see Section 4.2), L-S was retained, as it allowed for the analysis of irregular sampling regimes and for its convenience in processing extremely short sequences.[7]

2.2.2 Shannon Sampling Theorem

The Shannon Sampling Theorem, in its original formulation[8], states that:

> If a function $f(t)$ contains no frequencies higher than W cps[9], it is completely determined by giving its ordinates at a series of points spaced $\frac{1}{2}W$ seconds apart [23].

In practical terms and more modern vocabulary, this means that the sampling rate for a uniformly sampled signal must be greater than twice the highest frequency component of a bandlimited signal in order to guarantee complete capture of the

[6]The details of the mathematics of the L-S method and its relation to other LSSA techniques is beyond the scope of this document.

[7]The L-S periodogram can be used to generate spectra with arbitrary resolution, independent of the number of samples in the time domain. While there is no actual increase in information present, this does make it much easier to quickly compare results from sequences with differing number of samples, especially at low sample numbers.

[8]This phrasing contains a subtle inaccuracy: the frequency content of the signal must be strictly less than W, not less than or equal to.

[9]Cycles per second, a unit of frequency equivalent to Hertz (Hz).

signal. This requirement is also known as the Nyquist criterion. Mathematically,

$$f_c = \frac{f_s}{2} \tag{2.1}$$

and

$$f_{max} < f_c \tag{2.2}$$

where

f_c is the critical frequency[10]

f_s is the sampling frequency

f_{max} is the highest frequency present in the signal of interest.

Equation 2.1 defines the Nyquist limit for a given sampling rate, and Equation 2.2 defines its relationship to the frequency content of the sampled signal.

2.2.3 Aliasing

If the Nyquist criterion is violated, signal corruption can result. This corruption normally manifests as aliasing, in which signal content above the Nyquist limit in the original signal appears as lower-frequency content in the sampled signal.

[10] Also known as the Nyquist frequency or Nyquist limit.

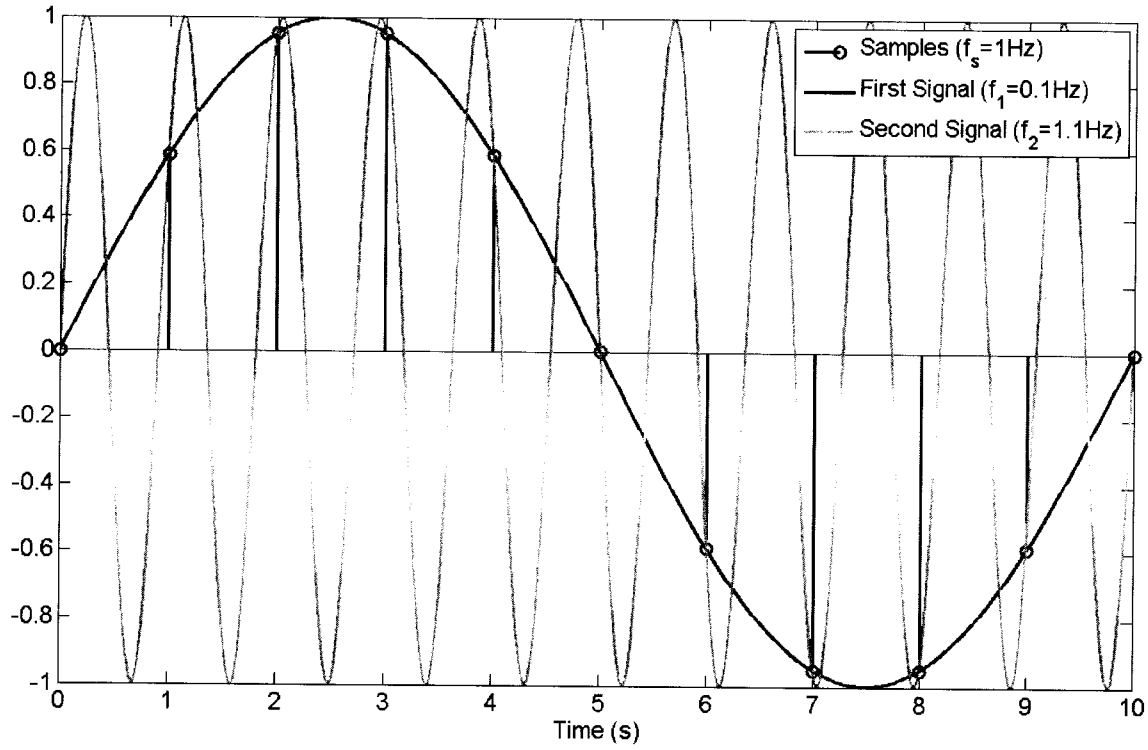
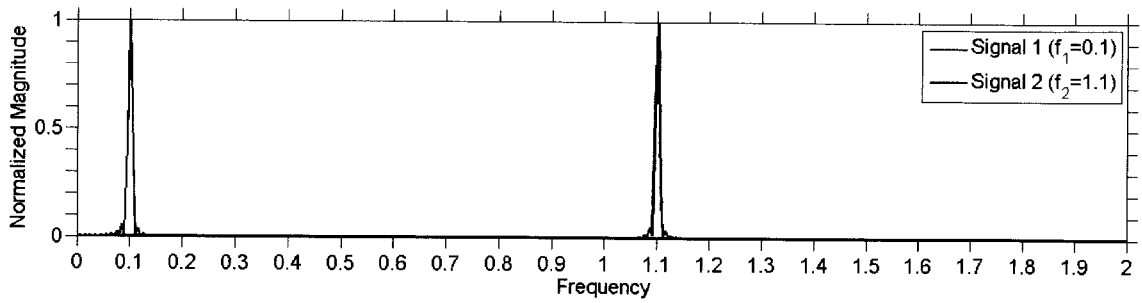


Figure 2.5: Example of aliasing: two signals, one of which aliases to the same frequency and sample values as the other.

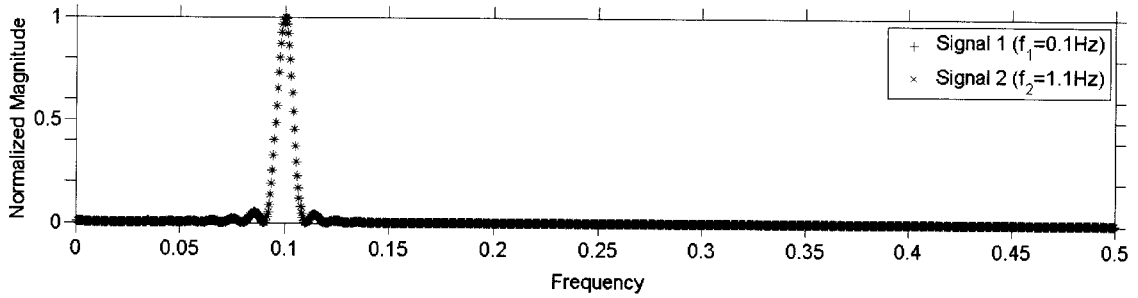
Figure 2.5 illustrates this phenomenon: the high frequency signal coincides with the low frequency signal at each of the sampling instants. The variation in the high frequency signal is not captured by the samples, and so it appears to be identical to the low frequency signal. The resulting effect in the frequency domain is shown in Figure 2.6. Note, particularly, that the spectra of the two signals exactly overlap in Figure 2.6(b).

2.2.4 Low-Pass Filtering

The most common technique for preventing aliasing is to Low-Pass Filter (LPF) the signal before sampling it. Removing all signal content at or above the Nyquist



(a) Spectra of both signals with $f_s = 10$ Hz



(b) Spectra of both signals with $f_s = 1$ Hz

Figure 2.6: Spectra of signals 1 and 2 from Figure 2.5 when (a) oversampled and (b) undersampled.

frequency[11] guarantees that aliasing will not occur (given uniform sampling), since the sampled signal is now band-limited in accordance with Equation 2.2.

Unfortunately, LPF is not a viable technique in TSGES since the signal must be filtered as a continuous time signal before sampling. The only currently available methods of measuring mRNA levels are inherently sample-based, which makes it impossible to LPF the signal.

The one possible exception would be to LPF the biological processes themselves, which would disrupt the context of the study so severely as to render the entire exercise pointless. Directly manipulating the GRN and its regulatory mechanisms to force its variations below the Nyquist limit of the chosen sampling rate would prevent aliasing, but would also mean that the results would not be relevant to the original,

[11] In practice, the filter is normally designed to remove all content above a set-point which is slightly *lower* than the Nyquist frequency.

unmodified organism.

2.3 Alternative Methods For Optimum Sample Selection

A careful review of the extant literature revealed only one other attempt to determine an optimal approach to sampling in TSGES. The foundations of this technique are given in [24], which presents a method for interpolation between sample points in a TSGES and generating a smooth, continous curve from the discrete data points.

The same researchers, based in part on the interpolation technique in [24], proposed a clever way of validating gene expression profiles. Taking the definite integral of the generated continuous curve over the interval defining the experiment and dividing by the length of the interval gives a value which should represent the overall average level of expression for a given gene during the experiment. Physically mixing (frozen or otherwise preserved) samples from each time point in the experiment and then measuring them using a microarray likewise provides a measure of the average level of expression of each gene. The correspondance between these two values gives an indication of the quality of the measured data points, while only requiring one additional microarray [25].

Finally, they presented, in [26] and [27], an algorithm for identifying near-optimal sample time points. Their method depends on the fact that while it is very expensive to measure the gene expression levels of a sample, the cost of taking and preserving a biological sample is negligible. In their proposed method, an extremely large number of samples are taken and stored during the TSGES experiment but not measured. Instead, initially only a very small number of samples are measured using microarrays (typically, less than one-half of the number of microarrays budgeted for are used

in this step). Based on this sparse data set, a measure of the "smoothness"[12] of the (interpolated) data is calculated, and the least "smooth" portion of the curve identified. The previously-taken but unmeasured sample in the center of this "rough" section is then analyzed and added to the profile, and a new smoothness meaure calculated. This repeats iteratively until the allocated number of microarrays has been expended.

[12]The paper suggests a particular measure of smoothness, but the algorithm is designed such that this measure can be easily replaced by one of the researcher's choosing.

Chapter 3

Aliasing in a Simulated Gene Regulatory Network

In order to easily evaluate the effectiveness of aliasing-mitigation techniques in biological networks, it is first necessary to establish a GRN model system in which aliasing can be shown and measured. The GAL regulon, a GRN which serves much the same role in *S. cerevisiae* as the *lac* operon does in *E. coli*, is ideal for this purpose. In this chapter, the GAL regulon is introduced (Section 3.1), an Ordinary Differential Equation (ODE) model (Section 3.3) and a software simulation thereof (Section 3.4) are presented, and this simulation is used to demonstrate that aliasing can occur in a GRN (Section 3.5); mitigating this and similar aliasing is the target of the techniques presented later in this work.

3.1 The GAL Regulon

The GAL regulon is a GRN which regulates the metabolism of galactose in *S. cerevisiae*. The function of the GAL regulon is similar to that of the *lac* operon (Section 2.1.5), in that it turns on or off the production of enzymes required to metabolize a particular sugar (galactose) only when that sugar is present and preferred sugars

(such as glucose) are absent. This logic is represented graphically in Figure 3.1, analogously to Figure 2.2.

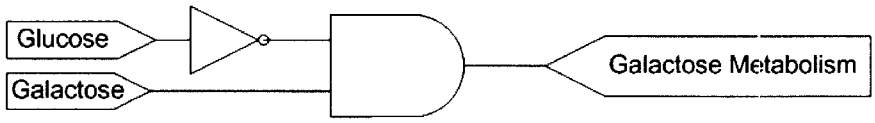


Figure 3.1: A digital logic gate representation of the GAL regulon.

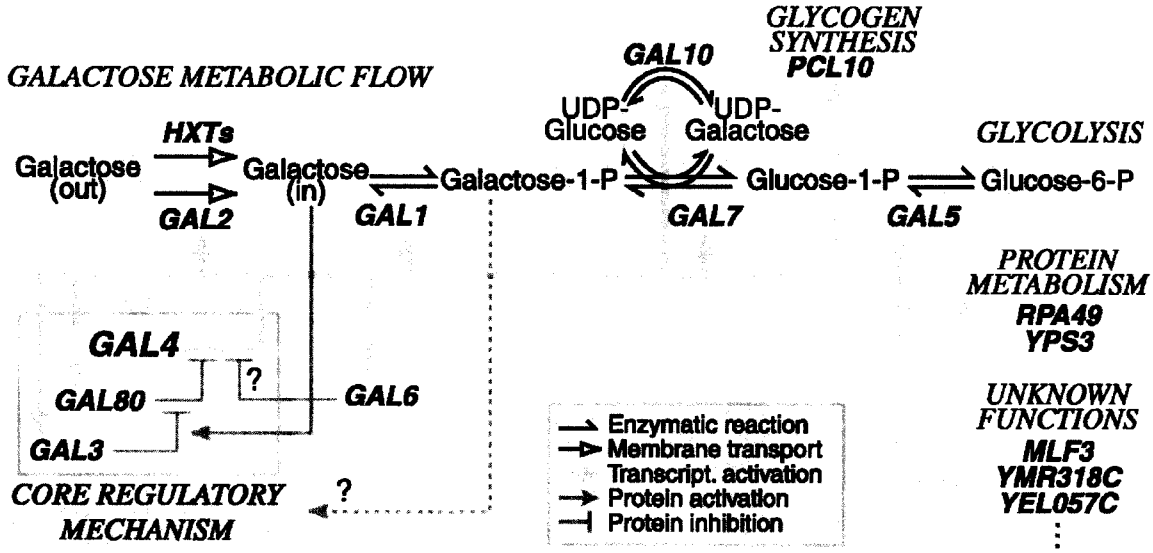


Figure 3.2: Metabolism of galactose in *Saccharomyces cerevisiae* (*S. cerevisiae*). This depiction is slightly different from the simulated model, due to evolving knowledge of the GAL regulon.
Image credit: [28]

The view given in Figure 3.2 presents a high-level overview of the functionning of the GAL regulon. The main path of galactose metabolism procedes from left to right, indicated by the grey arrow. The protein GAL2 moves galactose from outside the cell to inside, where a sequence of reactions catalyzed by GAL1, GAL5, GAL7 and GAL10 convert galactose to glucose so that it can be metabolized. The production of these proteins is driven by GAL4, whose production is in turn suppressed by GAL80. Normally, therefore, GAL80 suppresses GAL4, and galactose metabolism does not

occur. When galactose is present it, in combination with GAL3, suppresses GAL80, permitting the production of GAL4, and the consequent production of the galactose metabolism enzymes.

3.2 Choice of System

The GAL regulon is useful for two reasons: first, its complexity (19 interacting species) is sufficient to make it an interesting model system without being overwhelming; second, it is extremely well-characterized. Most of the available models of GRNs are qualitative or, at best, semi-quantitative. No other fully quantitative models were found for other biological systems; the ODE form of this model is particularly convenient for simulation.

3.3 Model

Table 3.1 provides a summary of the species in the GAL regulon, and the links between the species are shown in Figure 3.3.

Element	Description	Type
R1	GAL1	mRNA
R2	GAL2	mRNA
R3	GAL3	mRNA
R4	GAL4	mRNA
R80	GAL80	mRNA
Rrep	Reporter	mRNA
G1	Gal1p	Protein
G2	Gal2p	Protein
G3	Gal3p	Protein
G3i	Gal3p (activated by galactose)	Protein
G4	Gal4p (monomer)	Protein
G4d	Gal4p (homodimer)	Protein
G80	Gal80p (monomer), nucleus	Protein
G80C	Gal80p (monomer), cytoplasm	Protein
G80d	Gal80p (homodimer), nucleus	Protein
G80Cd	Gal80p (homodimer), cytoplasm	Protein
Grep	GFP reporter	Protein
C3i;80	Complex of Gal3p(activated) and Gal80Cd	Protein
Gic	Intracellular galactose	Sugar

Table 3.1: The species present in the GAL regulon model.

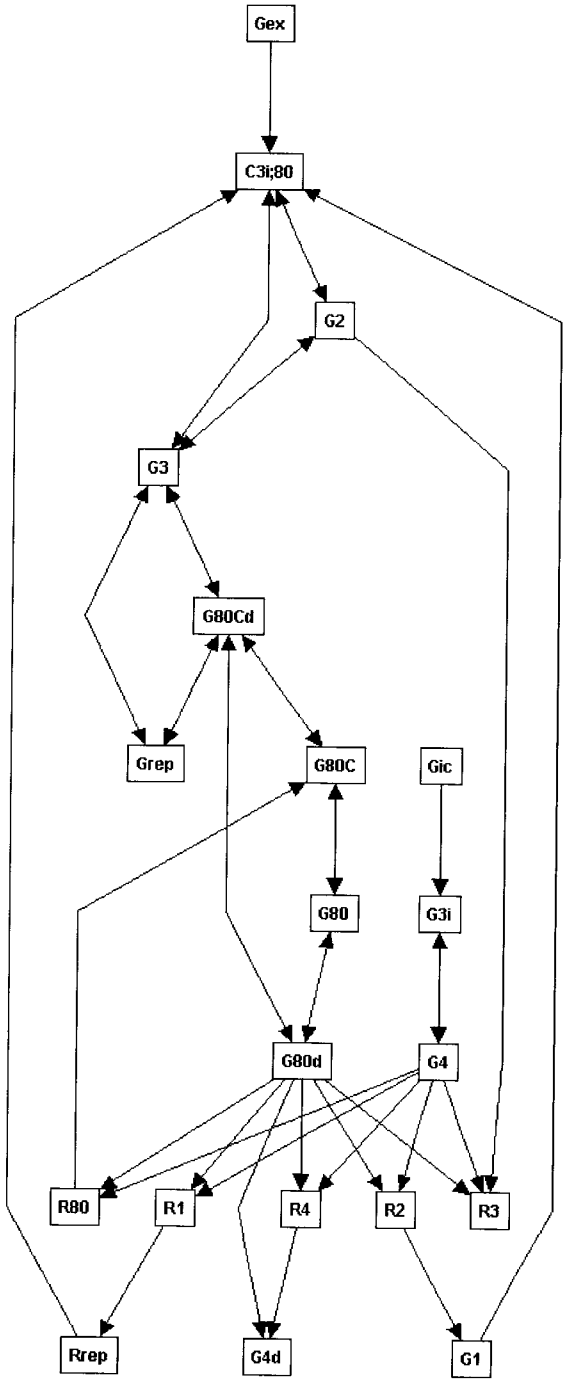


Figure 3.3: Interconnections in the GAL regulon model. The diagram has been simplified by supressing self-links and depicting reciprocal pairs of links as a bidirectional link.

3.4 GalSim

A complete ODE model for the GAL regulon, presented in [29] and refined by [30], is given in Appendix A.

The model described in Section 3.3 was implemented as a MATLAB[1] program referred to as GalSim (including code from [30]). Complete source code for the GalSim simulator is presented in Appendix B. In this implementation, the MATLAB Runge-Kutta based function `ode45` is used to evaluate the ODE model is contained in Listing B.7 based on the initial values in Listing B.5.

The model was supplied as a MATLAB code file containing the ODEs and a set of initial values. This author developed the necessary MATLAB code to exercise the model under different conditions, test TASS and jitter sampling options, analyze the resulting data and display formatted graphs of the most useful statistics.

Typical results are shown in Figure 3.4. The data were generated assuming that galactose was alternatingly present and then absent on a ten minute cycle with a 50% duty cycle (the same input parameters were used in all plots shown in this chapter). Individual curves in the figure correspond to some of the different species in the model.

Different species in the model show markedly different responses. Some of the signals (especially G_{80Cd} and R_{rep}) appear to vary on a ten minute cycle, matching the driving galactose input. Other signals appear to be nearly flat. This is to be expected, as the GAL regulon normally operates as a bistable switch, seeking one of two steady states depending on the extracellular concentration of galactose. The signals which show more variation are those which are more closely connected to the external input (see Figure 3.3). While this would initially appear to threaten the pertinence of the results, more detailed consideration suggests that this concern is unfounded. The GAL regulon system operates primarily by protein based promotors and inhibitors

[1]MATLAB Version 7.7.0.471, R2008b. The MathWorks, Inc., 3 Apple Hill Drive, Natick, Massachusetts, United States.

(see Section 2.1.4), which are amongst the slowest of regulation mechanisms; much faster mechanisms, such as siRNA, can be found in other GRNs. Likewise, other GRNs do not tend to steady state, and therefore show much more variability without forced oscillation. The combination of these two factors strongly suggest that other GRNs can be expected to show equal or greater rapid variation.

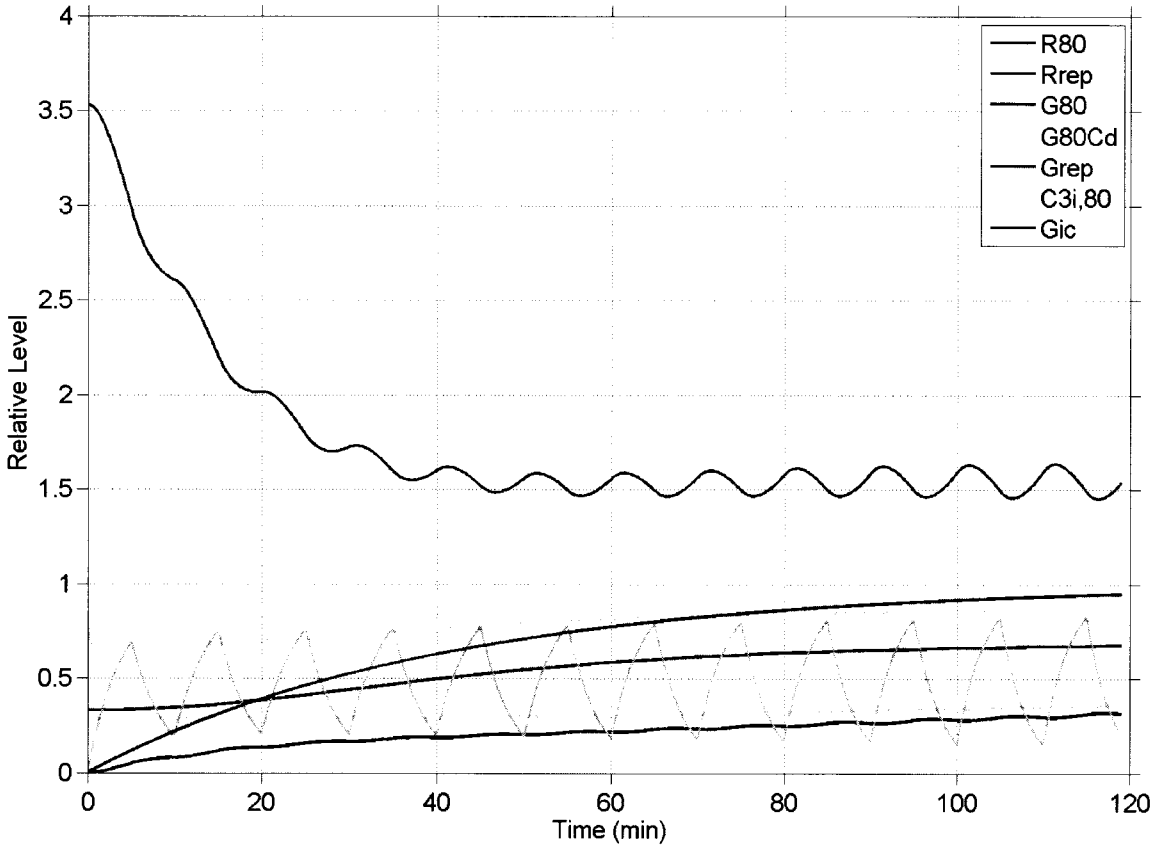


Figure 3.4: Time-domain GalSim results. See Section 3.4.1 for details of the presentation of the data.

3.4.1 Presentation of Results

As the various species present in the GalSim model are present in extremely divergent concentrations, it is convenient to normalize simulation results to a fixed standard so that they may be easily plotted on the same set of axes. Initial (starting) or

Element	Value
R1	0.92078
R2	1.14705
R3	2.73595
R4	0.92093
R80	3.59365
Rrep	460.36589
G1	4014.49353
G2	12597.78544
G3	19.64909
G3i	0.15652
G4	308.87968
G4d	460.56156
G80	0.14030
G80C	0.21727
G80d	44.58130
G80Cd	44.54901
Grep	1035.84459
C3i;80	1863969.18433
Gic	0.39992

Table 3.2: Normalization values used with the GAL regulon model.

steady-state values are useful for this purpose. The selected normalization values (which correspond to the initial value when non-zero, and the steady-state value otherwise) are given in Table 3.2. These normalized values will be identified as "Relative Magnitude" in the following figures.

The GalSim model contains 19 different species, many with very similar expression profiles. A plot of all of these curves in a single figure is somewhat cluttered and not easily grasped. Accordingly, only a subset (R_{80}, R_{rep}, G_{80Cd}, G_{rep}, $C_{3i;80}$ and G_{ic}) of

the species is presented in the following figures.

3.5 Demonstration of Aliasing

Figure 3.5 shows the frequency distribution of the various elements in GalSim[2] (with the values normalized as per the previous section). The majority of the signal content is found in the low-frequency region, with a significant DC component, but some signal content can be seen in the higher frequencies. The Nyquist cut-off corresponding to one sample every seven minutes is shown; the relevance is explained below. This presentation of the data, while accurate, conceals the full range of some of the signals, due to not using the full range of the y-axis. In order to emphasize the range of intensity, the spectra can be replotted in a way that covers the full vertical range of the graph. The same data is presented, with different normalization (all signals are normalized to have a maximum value of 1.0), in Figure 3.6; while this normalization technique makes it more difficult to compare one scenario to another, it facilitates determination of the relative distribution of spectral power within a single scenario.

[2]It is possible that these curves are subject to aliasing, but the trend of the curve (asymptotically approaching zero) indicates the contribution of aliasing, if any, is negligible.

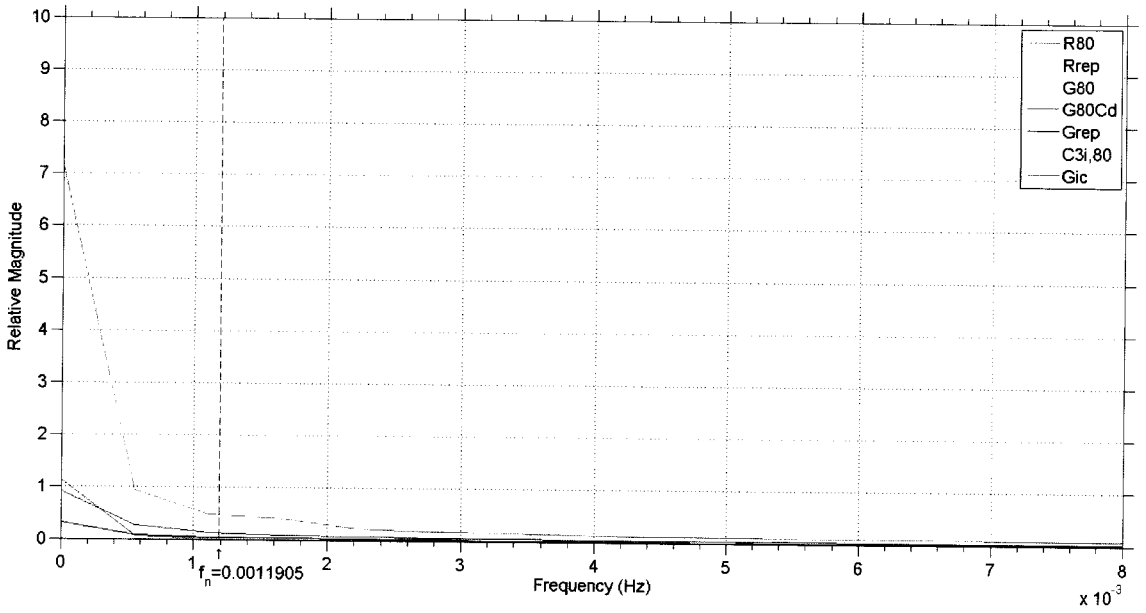


Figure 3.5: Spectrum from a GAL regulon simulation. Values are normalized as per Section 3.4.1. The Nyquist frequency for a 7 minute sampling interval ($f_n = 1.19 \times 10^{-3}$) is marked. Compare Figure 3.6 and Figure 3.7.

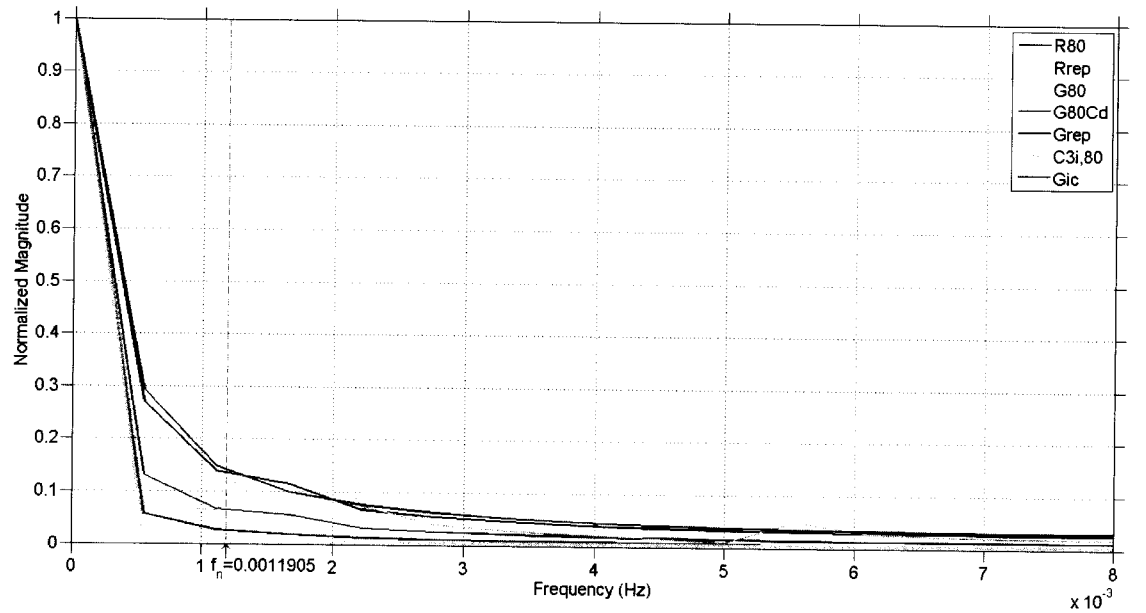


Figure 3.6: Spectrum from a GAL regulon simulation. Values are normalized such that maximum magnitude of each signal is unity. The Nyquist frequency for a 7 minute sampling interval ($f_n = 1.19 \times 10^{-3}$) is marked. Compare Figure 3.5 and Figure 3.7.

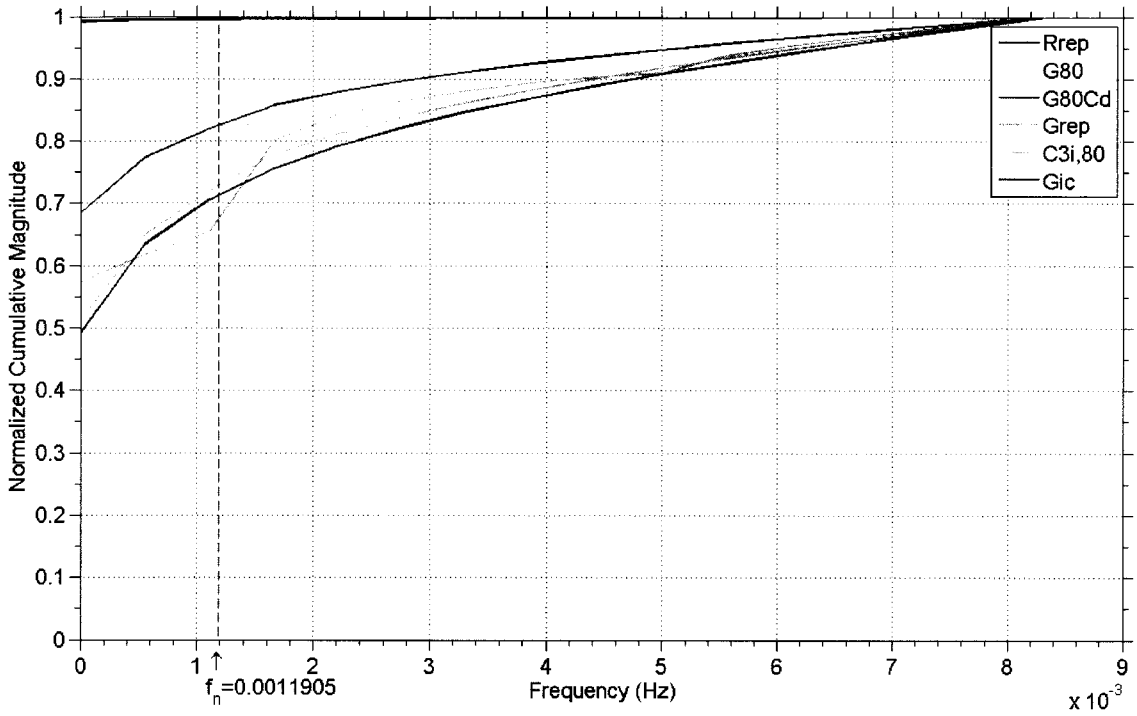


Figure 3.7: Cumulative spectrum from a GAL regulon simulation. Values are normalized such that maximum magnitude of each signal is unity. The Nyquist frequency for a 7 minute sampling interval ($f_n = 1.19 \times 10^{-3}$) is marked. Compare Figure 3.6.

An alternative depiction of the same data appears in Figure 3.7; in this presentation, the values are plotted as cumulative distributions. Equivalently, the curves can be described as giving the fraction of the total spectral content below the given point. The point at which a given curve crosses 0.5 on the y-axis partitions the signal into halves with equal spectral energy.

	Source	Duration (minutes)	# of Samples	Shortest Interval (minutes)
A	[8]	119	18	7
B	[9]	290	24	10
C	[10]	160	17	10
D	[31]	210	13	15

Table 3.3: A selection from the literature of sampling schemes used with *S. cerevisiae*, as per [32]. The letter indices correspond with Figure 3.8.

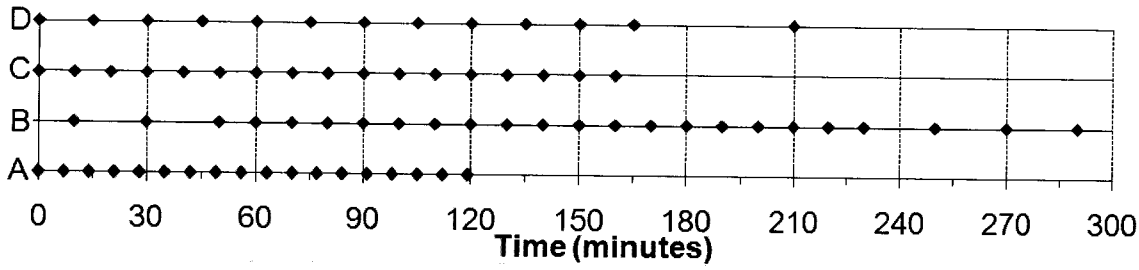


Figure 3.8: Sampling schemes used with *S. cerevisiae* in the literature, as per [32]. See Table 3.3 for the original sources.

A recent review paper [32] discussed the variation in sampling schemes used in TSGES. This data is reproduced in Table 3.3 and depicted graphically in Figure 3.8[3]. The shortest sampling interval, used in [8], is seven minutes. The Nyquist frequency (see Section 2.2.2) for this sampling rate is $f_c = \frac{1}{2} f_s = \frac{1}{2} \left(\frac{1}{420s}\right) = 1.19 \times 10^{-3}$ Hz, and is shown on the spectra plots above.

Examining Figure 3.7, the y-axis values at the specified cut-off frequency vary varies from 0.65 for $C_{3i;80}$ to near-unity for R_{rep}. If 65% of the signal content is below the Nyquist frequency, then $100\% - 65\% = 35\%$ of the signal content is above the Nyquist frequency. Given this, it can be stated with confidence that, under the given experimental conditions, 35% of the signal content of at least one species in the GAL regulon system sampled every seven minutes is aliased.

[3]The paper focused on and discussed studies of *S. cerevisiae*, not *E. coli*, but the cited data are typical for the study of most unicellular organisms.

3.6 Conclusions

The GAL regulon is a well-understood and useful model system for understanding and simulating gene regulation interactions. Simulations of the GAL regulon demonstrate that aliasing can occur in TSGES using an aggressive sampling interval taken from the literature. This is a novel finding: to date, the problem of aliasing in TSGES has not been expressly examined in the literature. This work is the first to demonstrate explicitly that aliasing does occur in this context.

Chapter 4

Jitter Sampling

Jitter sampling is a technique in which the time at which samples are taken is varied by small, random amounts. This technique is discussed extensively in [33]. For (non-jitter) periodic sampling with sampling interval T ($T > 0$), the set of sample time points can be defined as $\{t_n | t_n = nT, n \in \mathbb{N}, n > 0\}$. An analogous set of jitter sampling timepoints can be defined by

$$t_n = nT + \Delta t_n \tag{4.1}$$

where Δt_n is a set of random values taken from a given statistical distribution. The advantage of sampling according to Equation 4.1 is that jitter sampling can suppress aliasing of frequency components above the Nyquist frequency. Due to complexity of implementing jitter sampling, its primary use is in "processing signals that, due to technical or economical constraints, cannot be sampled fast enough to facilitate usage of classical DSP"[34]. This is consistent with TSGES, for which sampling rates are heavily constrained (see Section 2.1.3) and the signals cannot be made bandlimited (see Section 2.2.4).

4.0.1 Inversion of Timepoints

The definition given in Equation 4.1 allows for the inversion of timepoints; that is, if the distribution of Δt_n includes values exceeding one-half of the sampling interval T, it is possible for t_n to fall *after* t_{n+1}. For the purposes of this work, we disallow that possibility: any generated Δt_n which would result in an inversion is discarded and regenerated.

4.1 Applicability to TSGES

4.1.1 Wet-lab Implementation

Jitter sampling can easily be applied to TSGES. A computer program, given the nominal sampling time points (as would be used if jitter sampling were not to be employed) and the desired statistical properties of the jitter, can produce a listing of appropriately modified time points. Based on the experiments described herein, there does not appear to be any need for "strong" randomness in this application: a typical software-based pseudo-random number generator was used (see Appendix B for implementation details).

4.1.2 Unintentional Sources of Jitter

It may well be that jitter sampling is already occuring in TSGES, albeit unintentionally. While researchers are normally careful to adhere closely to the intended sampling times, discrepancies are inevitable.

Some potential sources of unintentional jitter include:

- errors in timekeeping on the part of the researcher
- variation in the effectiveness of the homogenization buffer

- variation in the time taken to transfer the sample to the homogenization buffer
- lack of synchronization between elements of the population being sampled
- uneven amplification of samples causing potentially out-of-sync cells to dominate the results for a particular sample

4.2 Comparison of FFT and L-S

The FFT assumes that all data points are evenly spaced, however, this is not the case when jitter sampling is used. As an alternate technique, L-S can be used to obtain the spectrum of irregularly sampled signals (see Section 2.2.1). This would, in principle, allow the evaluation of the jitter sampled signal based on the actual sample timepoints (as defined in Equation 4.1), instead of treating the samples as though they had been gathered without jitter.

We can define two sets of time points, with and without jitter. The first, which we call the "nominal" timepoints $\{T_{nominal}\}$, is defined to be set of timepoints used in a particular sampling scheme *without the application of jitter.* The second, the "actual" timepoints $\{T_{actual}\}$, is the sampling scheme timepoints after jitter is added. Note that, for any given sampling scheme with k (where $k \in \mathbb{N}$) samples, $k \equiv |T_{nominal}| \equiv |T_{actual}|$. Also note that matched any pair of corresponding timepoints $\{t_i = T_{nominal}[n], t_j = T_{actual}[n]\}$ are related by the corresponding jitter $\Delta t \equiv t_j - t_i$.

We can also define the set of jitter sampled values $\{V|V[n] = y(T_{actual}[n]); n \in \mathbb{N}, 0 \leq n \leq k\}$, where $y(t)$ is the synthetic data signal defined by Equation 4.3, consisting of the linear superposition of two sinusoids.

Three cases were evaluated, using 31 samples taken at 1 Hz (i.e., $T_{actual} =$

$\{0, 1, 2, 3, \ldots, 30\}$) with jitter values drawn from a Gaussian distribution with variance equal to ten percent of the nominal sampling interval. The three cases were:

- FFT of V ($T_{nominal}$ is implicit)
- L-S of $T_{nominal}$ and V
- L-S of T_{actual} and V

Each case was evaluated using the average spectrum from 10,000 runs.

For the FFT case, the sample array was zero-padded to a length of 2048 and was evaluated using the built-in MATLAB `fft` function.

Both L-S cases were evaluated using the implementation described in Section 2.2.1. The oversampling rate (`OFAC`, a parameter which governs the number of different candidate sinusoids which are tested against the) rate was selected as $\frac{2000}{31} = 64.5$, so as to give approximately the same number of data points as the FFT case.

It was found that the results for all three cases (after averaging multiple trials to compensate for statistical artifacts), as shown in Figure 4.1, are reasonably similar, aside from a moderately raised floor in the FFT case; this floor is believed to be the result of spectral leakage and is discussed furhter in Section 4.3.2. As the results are only compared to other scenarios evaluated using the same method, and not to any external values, it is believed that the discrepancy is of no consequence. For reasons of convenience, L-S with nominal timepoints was selected for the synthetic data model experiments.

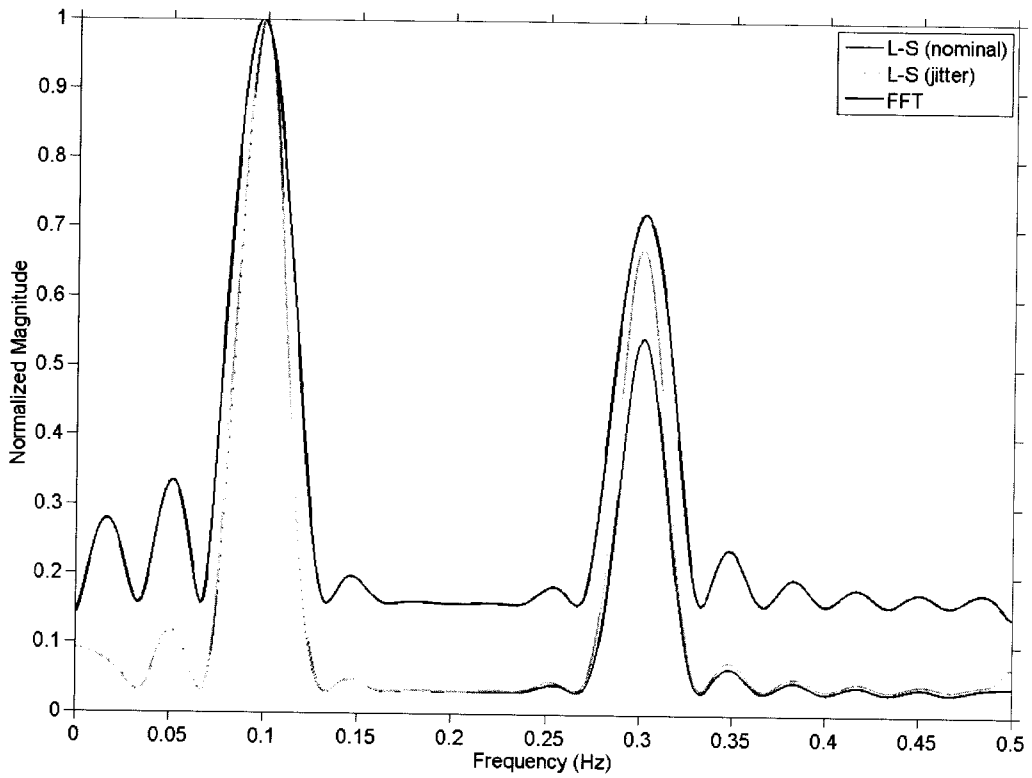


Figure 4.1: Results of different spectrum estimation techniques. The spectrum of the synthetic data model with jitter sampling (compare Figure 4.5) as determined using FFT, L-S with jittered time-points and L-S with non-jittered time-points. Results shown are the average of 10,000 trials.

4.3 Demonstration In Synthetic Data

4.3.1 Synthetic Data Model

Before applying jitter sampling to the GAL regulon model, it is helpful to test it in a much simpler system with easier to interpret results. The simplest system which is useful in analysing the effectiveness of alias suppression is a single signal formed by the linear superposition of two sinusoids, one above and the other below the Nyquist criterion for the sampling rate of interest. Mathematically, this can be described as

$$y(t) = A_1 \cdot \cos\left(2\pi f_1 \cdot t + \phi_1\right) + A_2 \cdot \cos\left(2\pi f_2 \cdot t + \phi_2\right). \tag{4.2}$$

For simplicity, we set $A_1, A_2 = 1$ and $\phi_1, \phi_2 = 0$, giving

$$y(t) = \cos(2\pi f_1 \cdot t) + \cos(2\pi f_2 \cdot t). \tag{4.3}$$

For this work, the frequencies $f_1 = 0.1$ Hz and $f_2 = 1.3$ Hz are used. The resulting signal (hereafter, the "synthetic data model") is shown in Figure 4.2.

Figure 4.3 shows the results of sampling this signal at 10 Hz rate (such that $f_1, f_2 < \frac{1}{2} f_s$). By contrast, Figure 4.4 shows the results when the sampling rate is reduced to 1 Hz (such that $f_1 < \frac{1}{2} f_s < f_2$). Note that the low frequency peak (at 0.1 Hz) is essentially unchanged, while the high frequency peak (originally at 1.3 Hz) is aliased to 0.3 Hz. Also note that spectral leakage is present, as expected for any finite-length frequency transform.

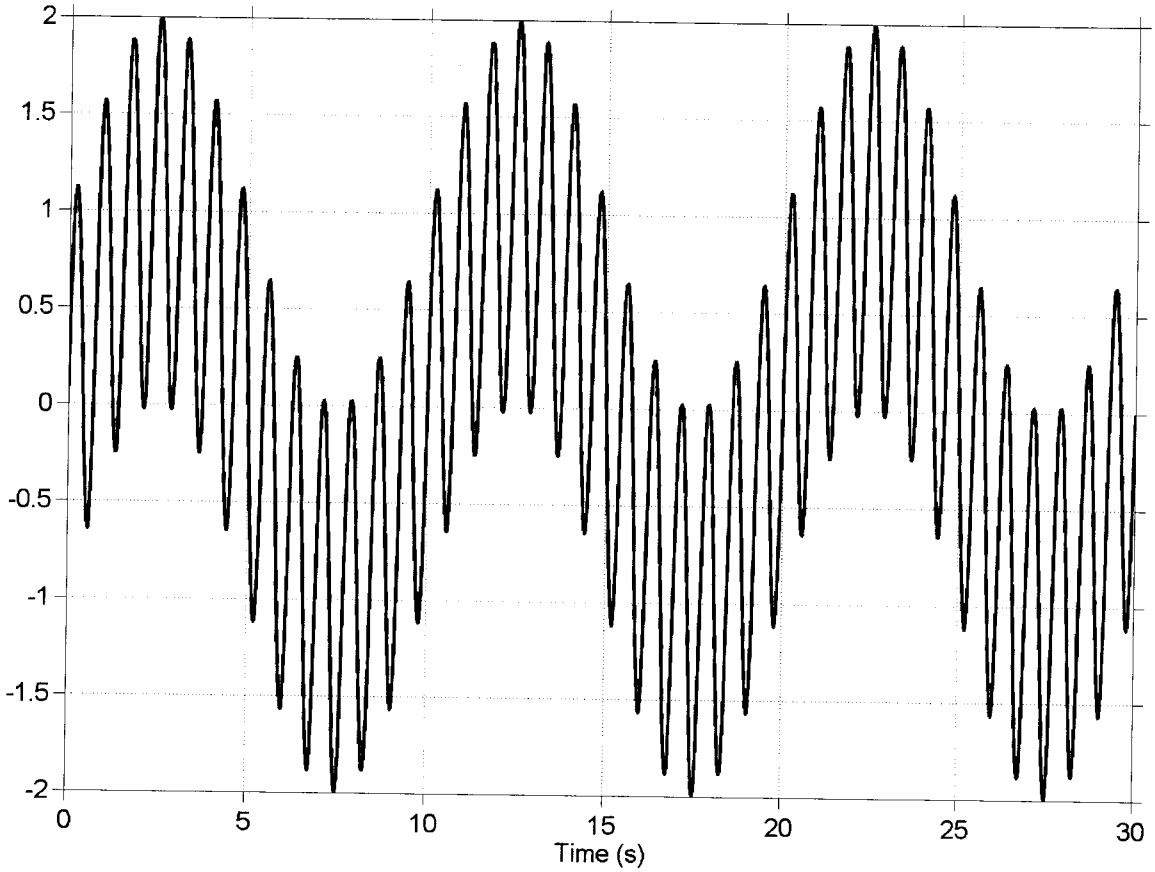


Figure 4.2: The time domain depiction of the signal used in the synthetic data model.

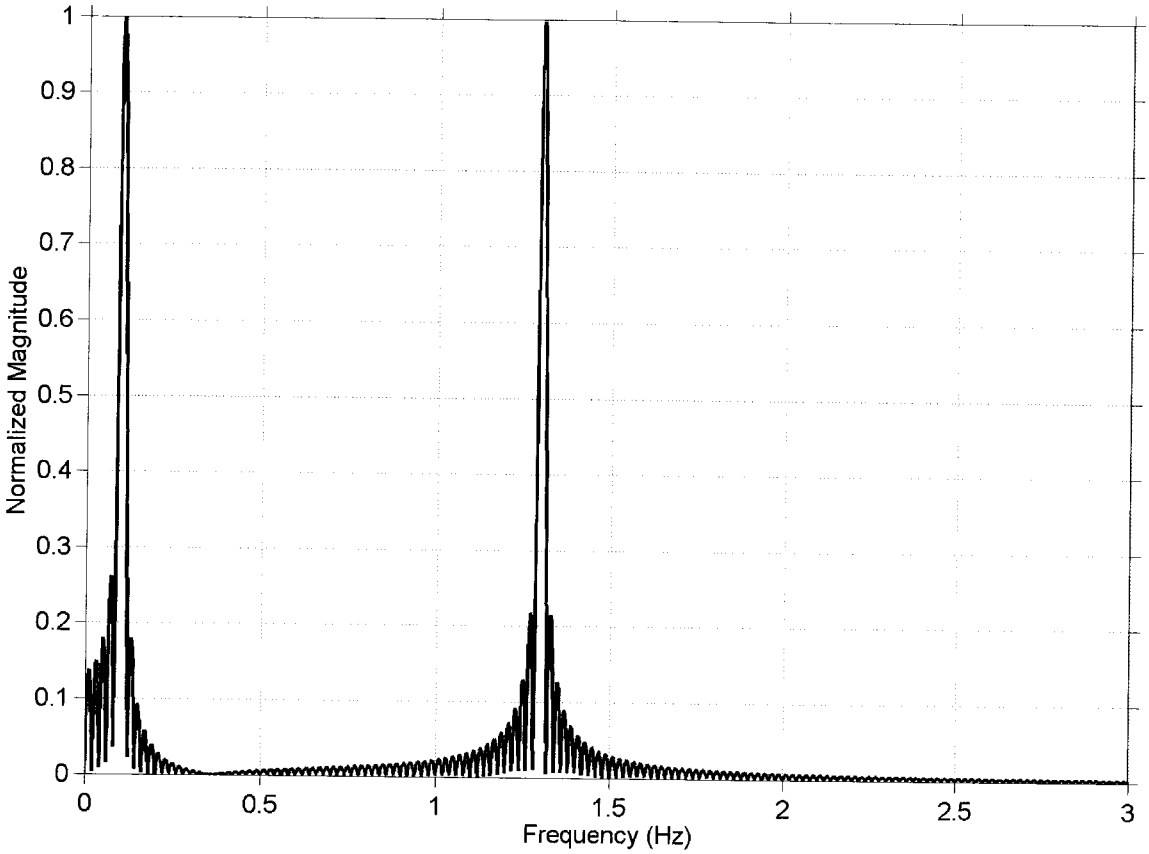


Figure 4.3: Oversampled spectrum of the synthetic data model, with $f_s = 10$ Hz.

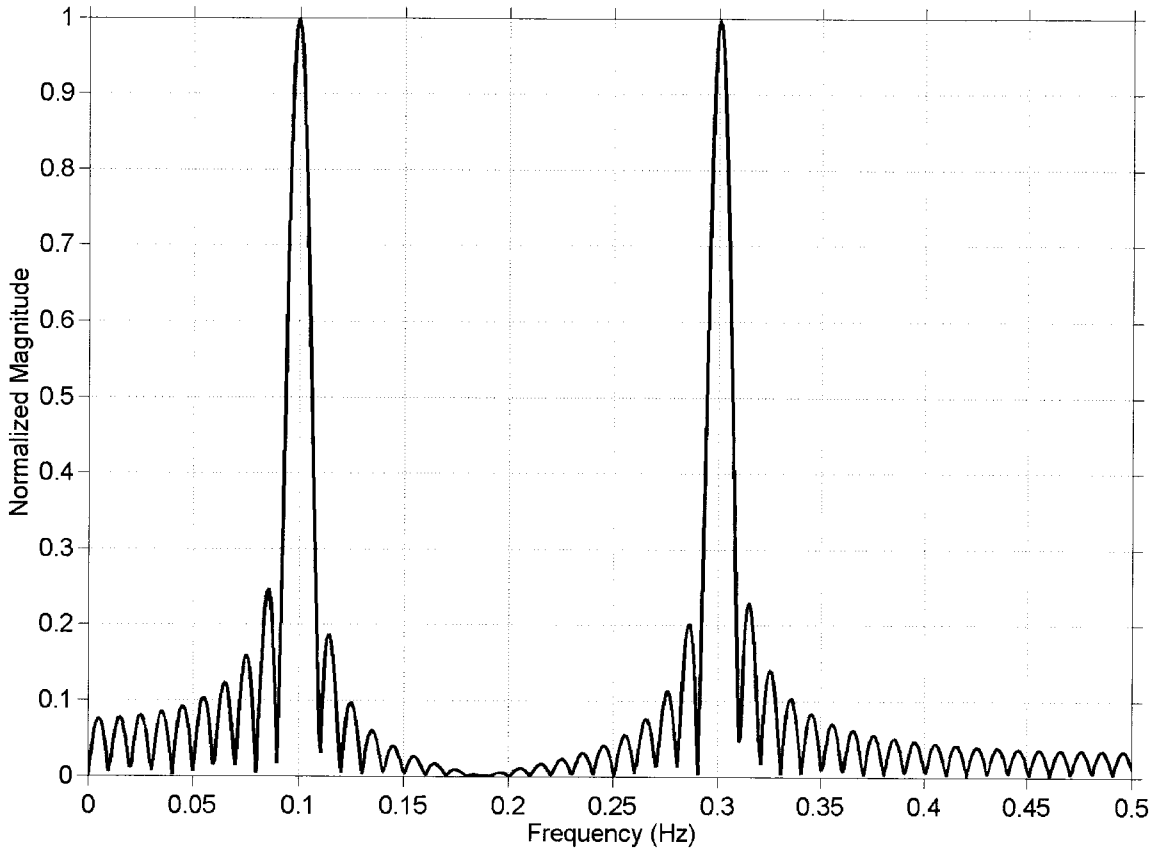


Figure 4.4: Undersampled spectrum of the synthetic data model, with $f_s = 1.0$ Hz.

4.3.2 Application of Jitter Sampling

The spectrum of the synthetic data model with jitter sampling applied is shown in Figure 4.5. The jitter was applied as per Equation 4.1, with Δt_n drawn from a Gaussian distribution with a variance equal to 10% of the non-jittered sampling interval.

Comparing Figure 4.5 with Figure 4.4 (note the decrease in magnitude of the aliased 0.3 Hz peak relative to the unaliased 0.1 Hz peak) gives a clear indication that jitter sampling can be effective in suppressing aliasing. It should also be noted that the off-peak frequencies show significant variation, including some mild increases; this is believed to be the result of the effect of jitter sampling on spectral leakage.

Spectral leakage (readily apparent in Figure 4.4) would be expected to be randomly "re-distributed" across the spectrum, producing the essentially unpredictable variations in the spectrum floor.

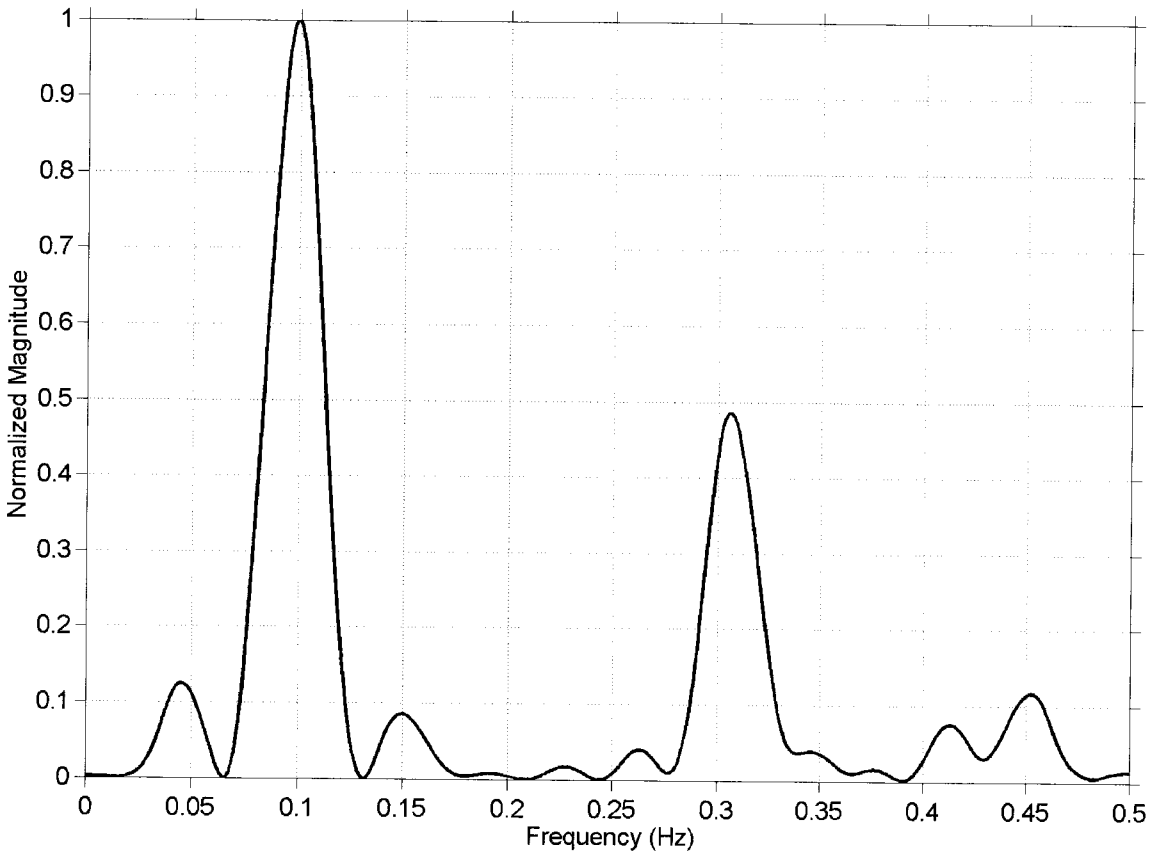


Figure 4.5: Synthetic data model spectrum with jitter sampling. The results shown are the result of a single trial, with the jitter constrained to a Gaussian distribution with a standard deviation of $\pm 10\%$ of the nominal sampling interval. Compare to Figure 4.4.

4.3.3 Measuring Alias Suppression

A simple, useful measure of the alias suppression effectiveness of jitter sampling in the synthetic data model is $Q = \frac{M_1}{M_2}$, the ratio of the magnitudes of the two spectrum peaks, as shown in Figure 4.6. Assuming both sinusoids have equal magnitude, $Q = 1$ if there is no alias suppression, and smaller values of Q indicate more effective alias

suppression.

In Figure 4.6, the aliased peak has a magnitude of $M_1 = 0.49$, and the non-aliased peak (by design) has magnitude $M_2 \equiv 1.00$. This gives an effectiveness of $Q = \frac{M_1}{M_2} = \frac{0.49}{1.00} = 0.49$.

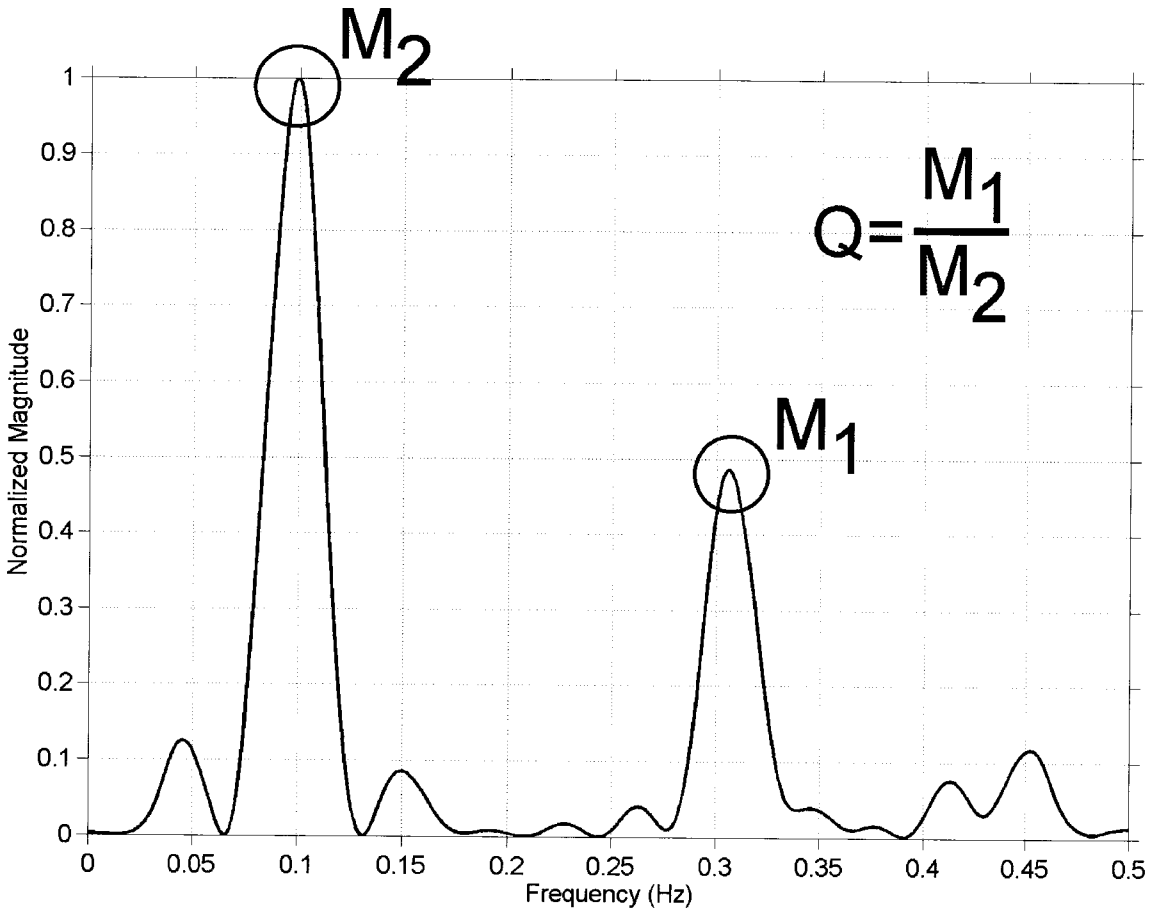


Figure 4.6: An illustration of a simple measure of alias suppression effectiveness. The measure is defined as the ratio of the magnitudes of the two spectrum peaks. Smaller values are better. In this example, $M_1 = 0.49, M_2 = 1.00 \Rightarrow Q = \frac{M_1}{M_2} = \frac{0.49}{1.00} = 0.49$.

This Q metric will be used in the following sections as a basis for comparing the degree of alias suppression in various scenarios.

4.3.4 Effect of Number of Samples

One of the important considerations in this work is that the number of samples available is at least an order of magnitude fewer than in more conventional DSP applications (e.g., telecommunications), and the quality of the signal interpretation declines with decreasing numbers of samples. It is therefore critical to determine how resilient the effect of jitter sampling is to low number of samples.

Figure 4.7 shows the measured spectrum of the synthetic model at a range of number of samples. This figure can be thought of as a "stack" of spectra, each one like the one shown in Figure 4.5, but using a different number of samples. This is done by generating a series of 200 samples taken at one second intervals from the synthetic data model (with the appropriate jitter sampling applied), and then calculating the frequency content after truncating the series to the desired length.

Evaluating each spectrum as per Section 4.3.3, and plotting number of samples versus Q, we can see that, as expected, alias suppression improves with increasing number of samples, as shown in Figure 4.8. Note, though, that this figure is based on a single run, and is thus subject to significant statistical artifact. The basic trend is consistent, and more obvious in Figure 4.10 through Figure 4.14, which average together multiple trials (these figures will be discussed in more detail in the next section). Also important is the fact that jitter sampling continues to have a noticeable effect with as few as ten samples - fewer than was used in any of the studies referenced in Table 3.3 from Section 3.5.

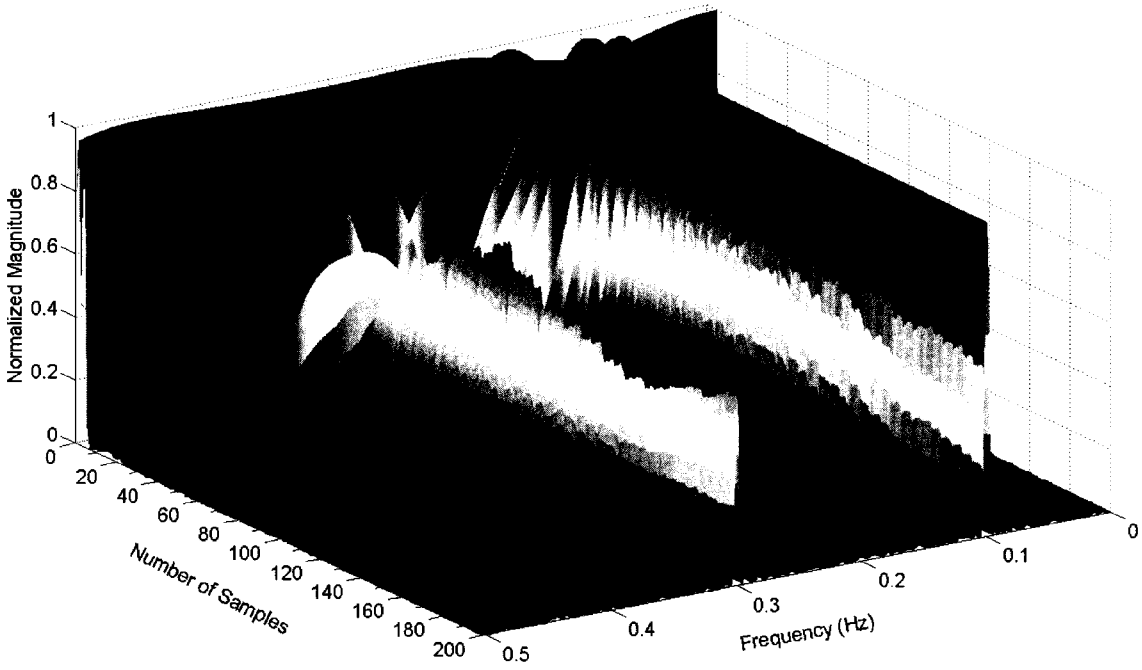


Figure 4.7: A three-dimensional plot of spectra at various sample numbers. The results shown are the result of a single trial, with the jitter constrained to a Gaussian distribution with a standard deviation of $\pm 10\%$ of the nominal sampling interval.

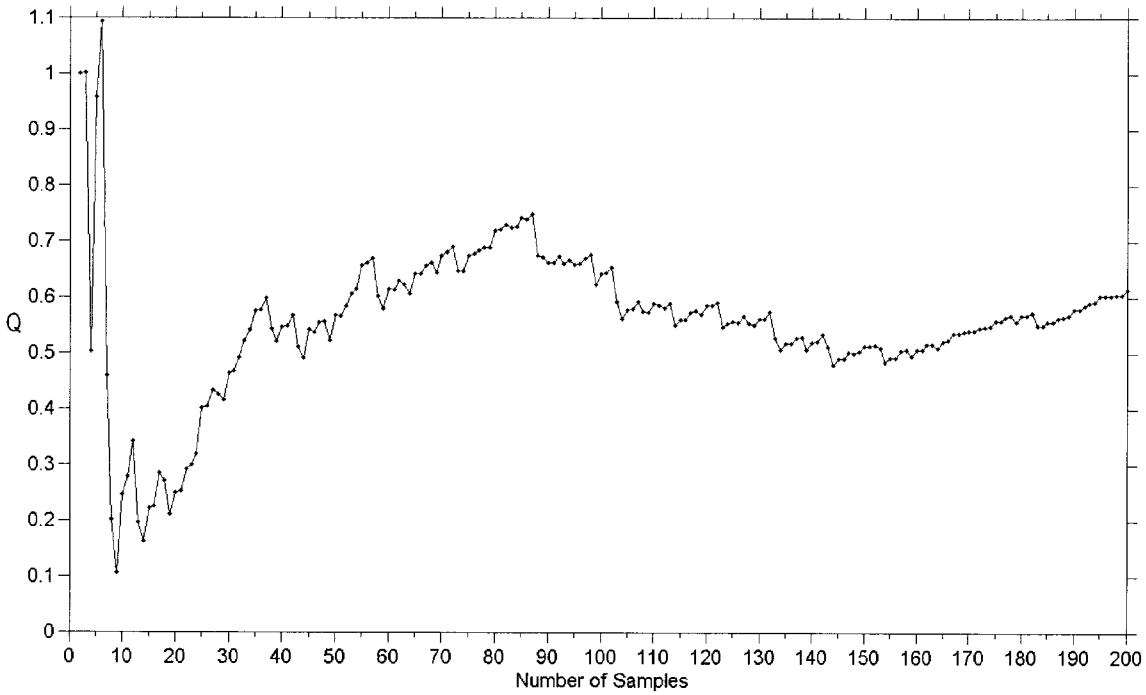


Figure 4.8: Alias suppression effectiveness of jitter sampling versus number of samples. The data shown are the Q values of the spectra shown in Figure 4.7. As this figure is the product of a single trial, significant stochastic artifact is present.

4.3.5 Effect of Jitter Distribution

The choice of statistical distribution from which the jitter values are drawn has an effect on the results of jitter sampling. Two main parameters govern this choice: the shape of the distribution and its variance.

Effect of Jitter Distribution Shape

The first item to be considered is the shape of the statistical distribution. While the distribution can theoretically take any shape, only two are considered in this work: Gaussian and uniform. Figure 4.9 shows plots of both functions, corrected to have variance of 1.0. The results of using jitter sampling with both shapes are shown in Figure 4.10. The use of uniform jitter distribution appears to result in a

very slight decrease in Q as compared to Gaussian jitter when more than 50 samples are considered. Between 10 and 50 samples, the difference is negligible (results for less than 10 samples do not appear to be meaningfully intelligible due to inability to discriminate the two peaks and should be ignored). These results suggest that, assuming equal variance, the shape of the jitter distribution has a minimal effect on the alias suppression effect.

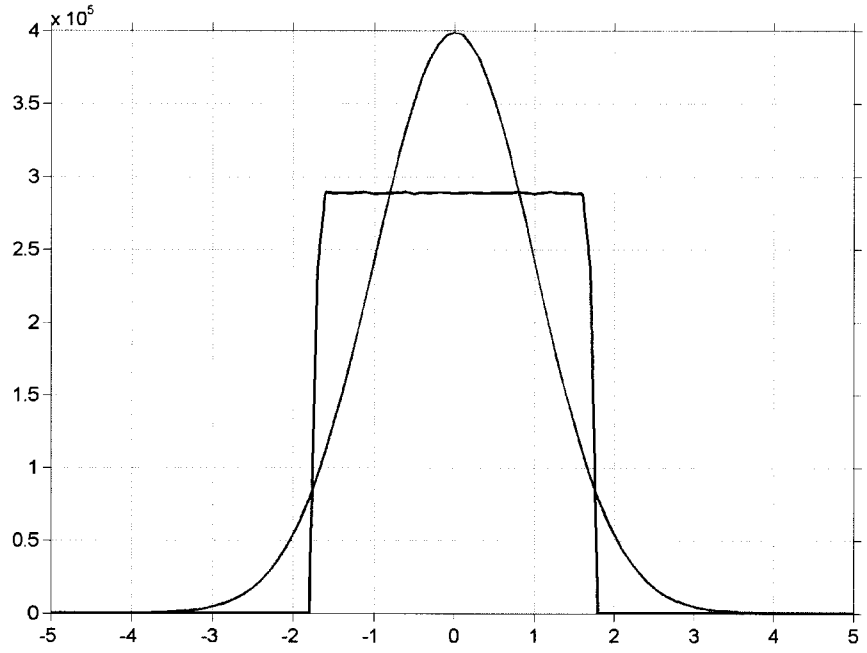


Figure 4.9: Gaussian and uniform statistical distributions. Both curves were generated using built-in MATLAB functions with a mean of 0 and a variance of 1.0.

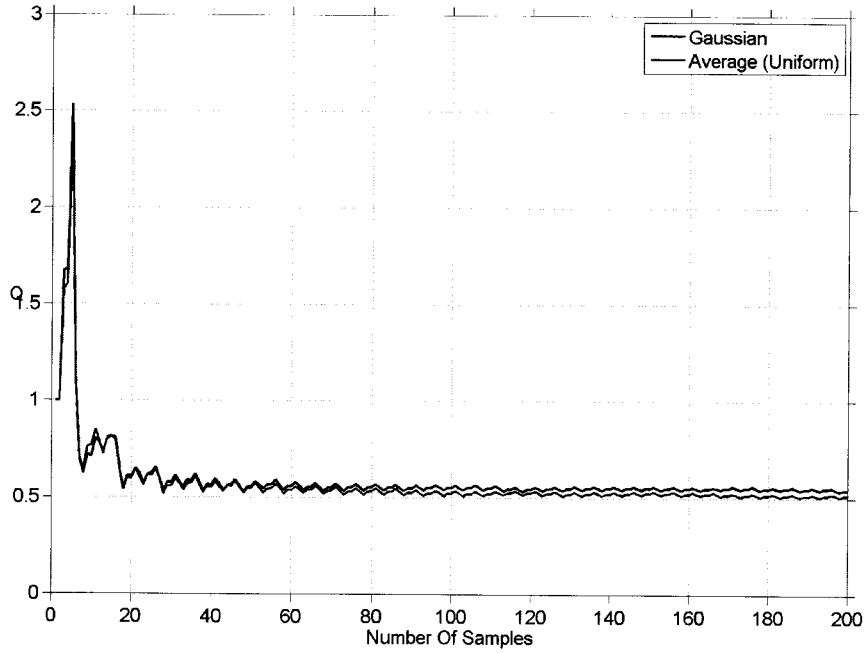


Figure 4.10: Alias suppression effectiveness of jitter sampling for both Gaussian and uniform jitter distributions. Results shown are the average of 100 trials.

Effect of Jitter Distribution Magnitude

The variance of the jitter distribution is also of great importance. Figure 4.11, Figure 4.12, Figure 4.13 and Figure 4.14 show the effect of increasing jitter variance, from 5% to 30% of the nominal sampling interval. A trend of increasing alias suppression with increasing jitter is evident, going from a Q of 0.8623 at 5% to 0.2432 at 30% (considered at the arbitrarily selected 30 sample mark). Again, results below 10 samples have little meaning due to lack of frequency domain resolution.

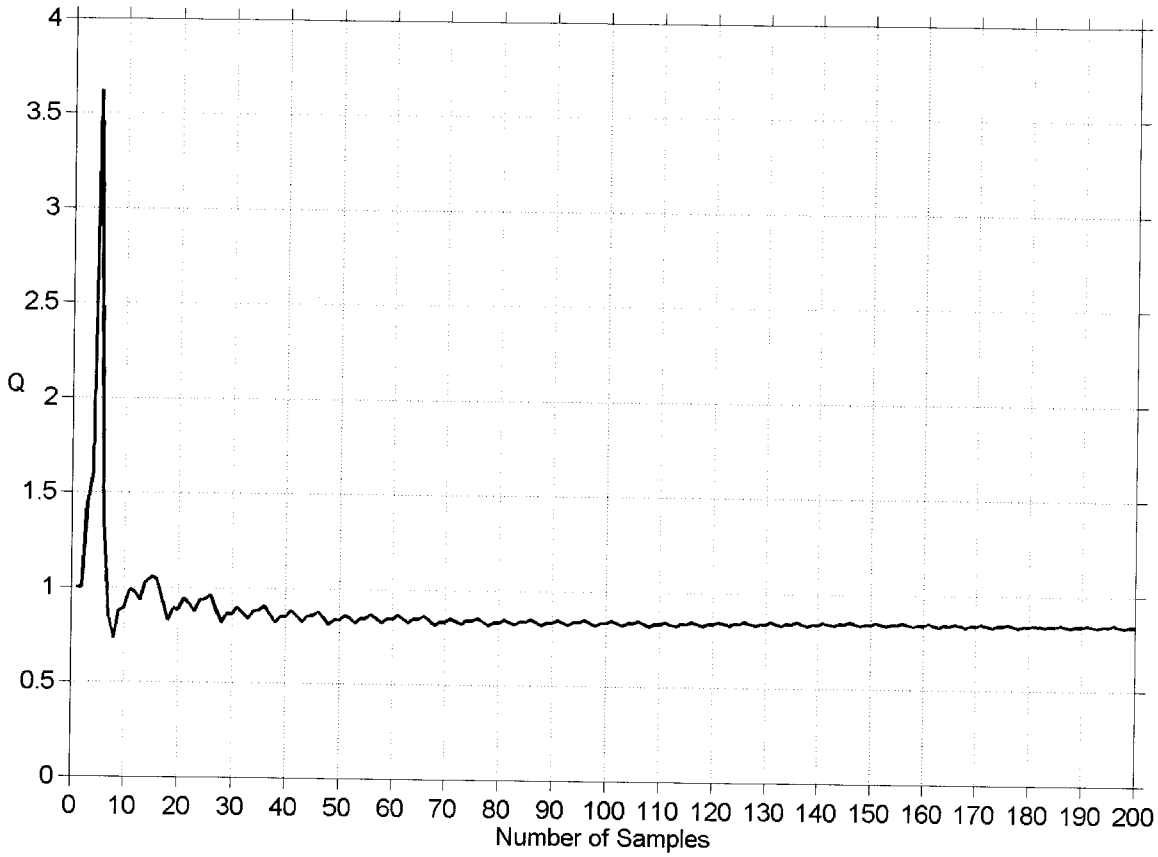


Figure 4.11: Alias suppression effectiveness of jitter sampling with 5% Gaussian jitter. Results shown are the average of 100 trials.

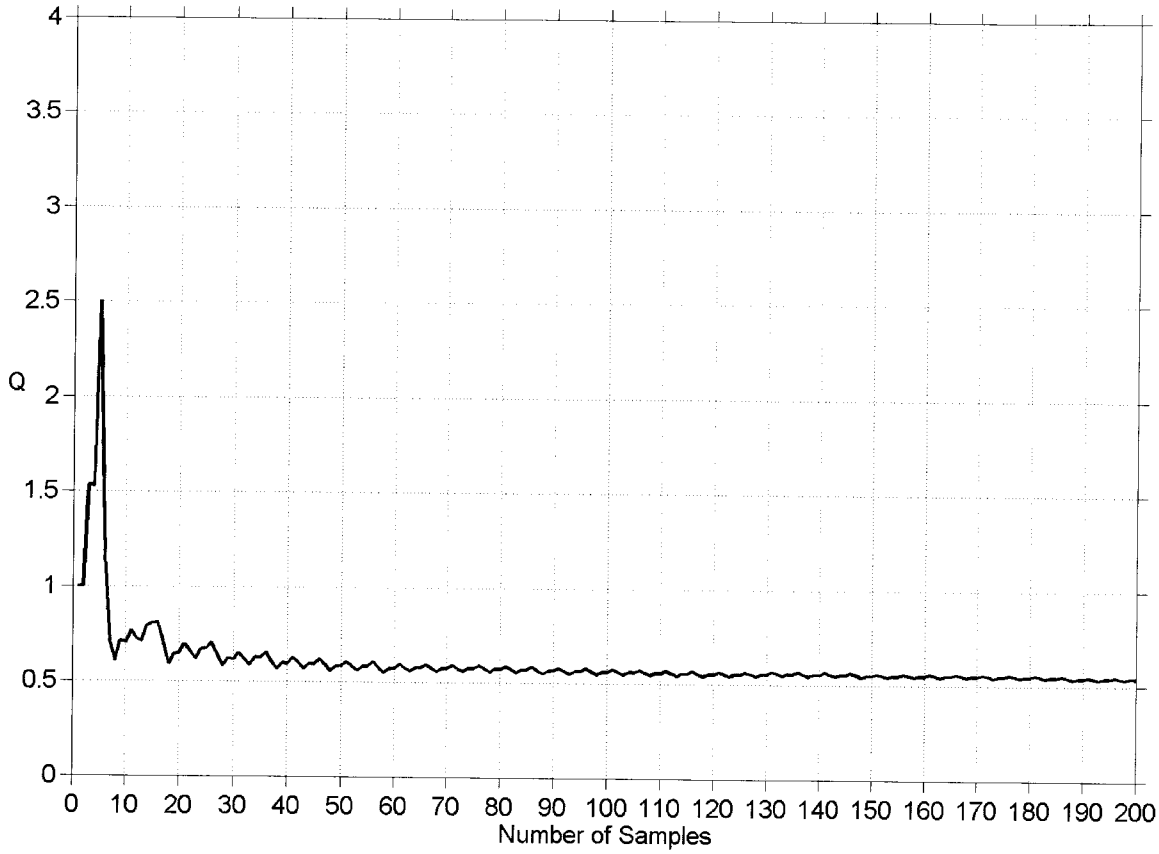


Figure 4.12: Alias suppression effectiveness of jitter sampling with 10% Gaussian jitter. Results shown are the average of 100 trials.

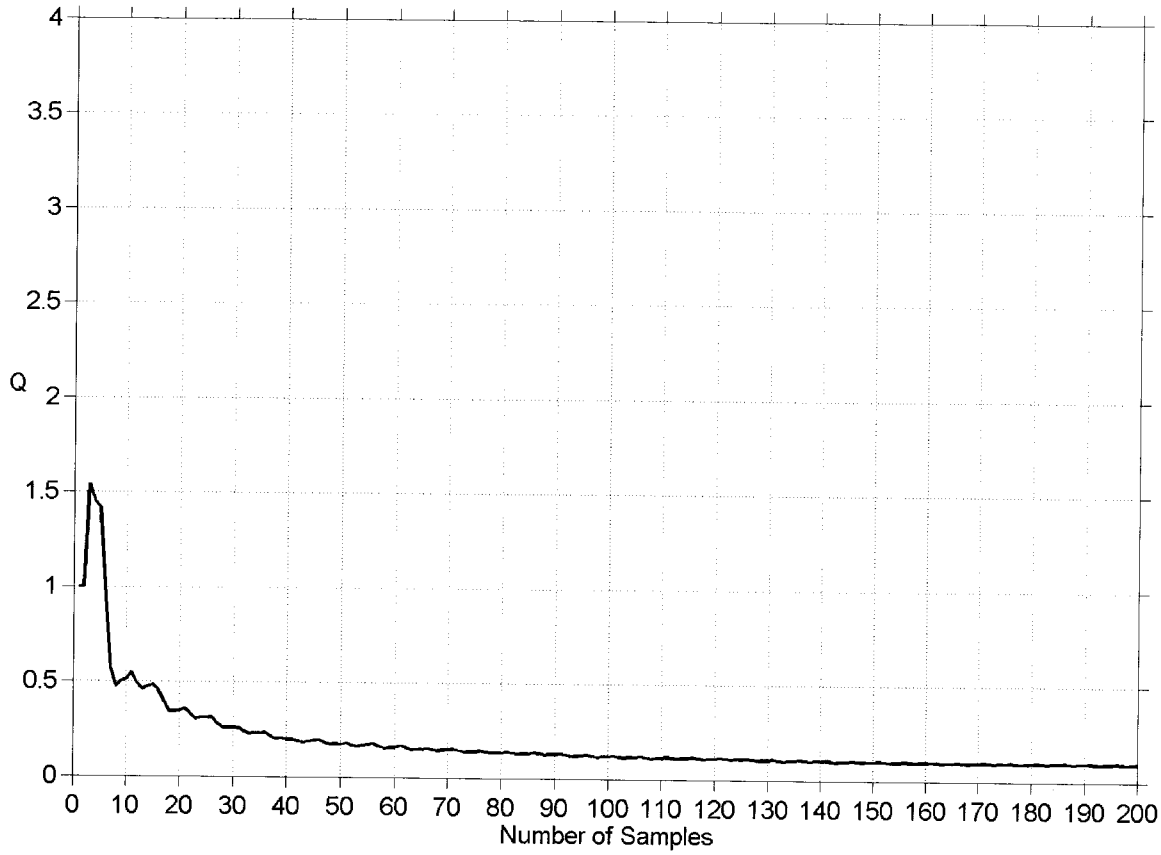


Figure 4.13: Alias suppression effectiveness of jitter sampling with 20% Gaussian jitter. Results shown are the average of 100 trials.

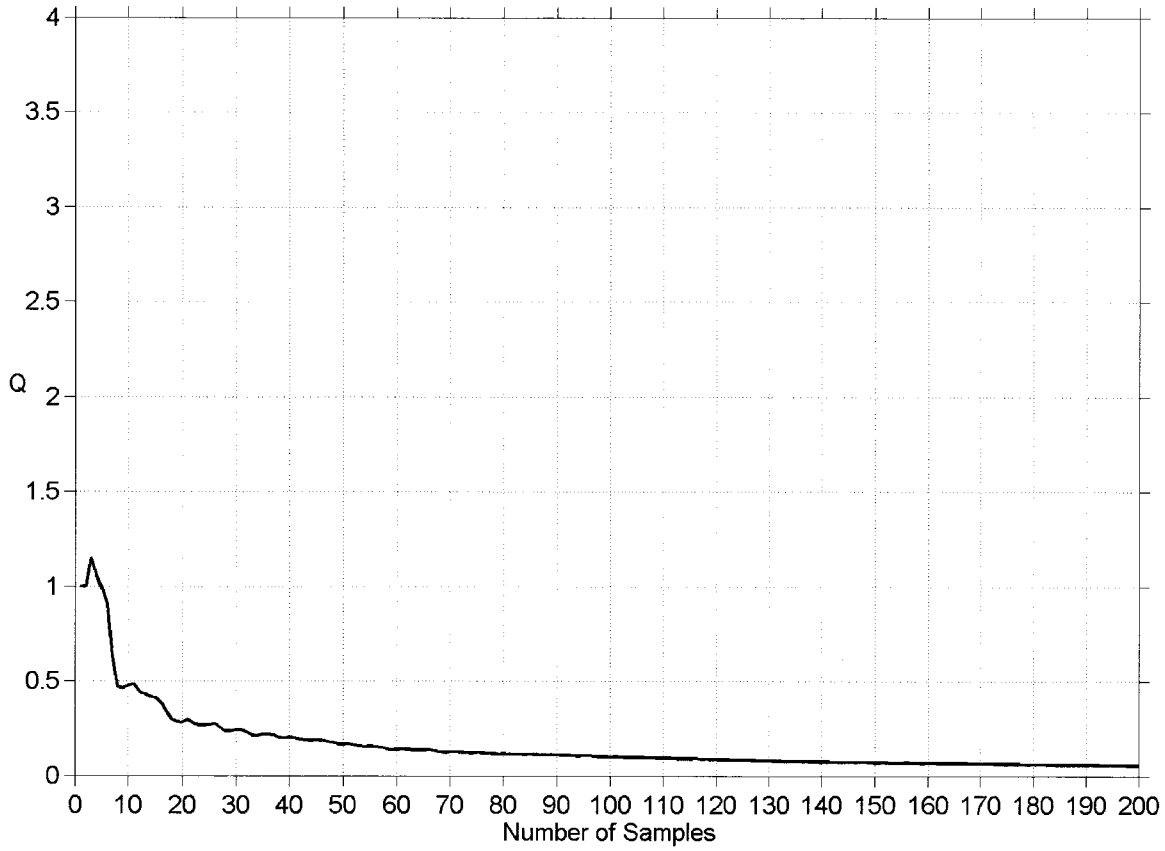


Figure 4.14: Alias suppression effectiveness of jitter sampling with 30% Gaussian jitter. Results shown are the average of 100 trials.

4.4 Variation in results

Since jitter sampling is by definition a stochastic process, the results vary from one trial to the next. The results plots shown in the previous sections are composites, showing the average of multiple trials. Even the pathological case where $\Delta t_n = 0 \forall n$ is possible; this is equivalent to not using jitter sampling at all.

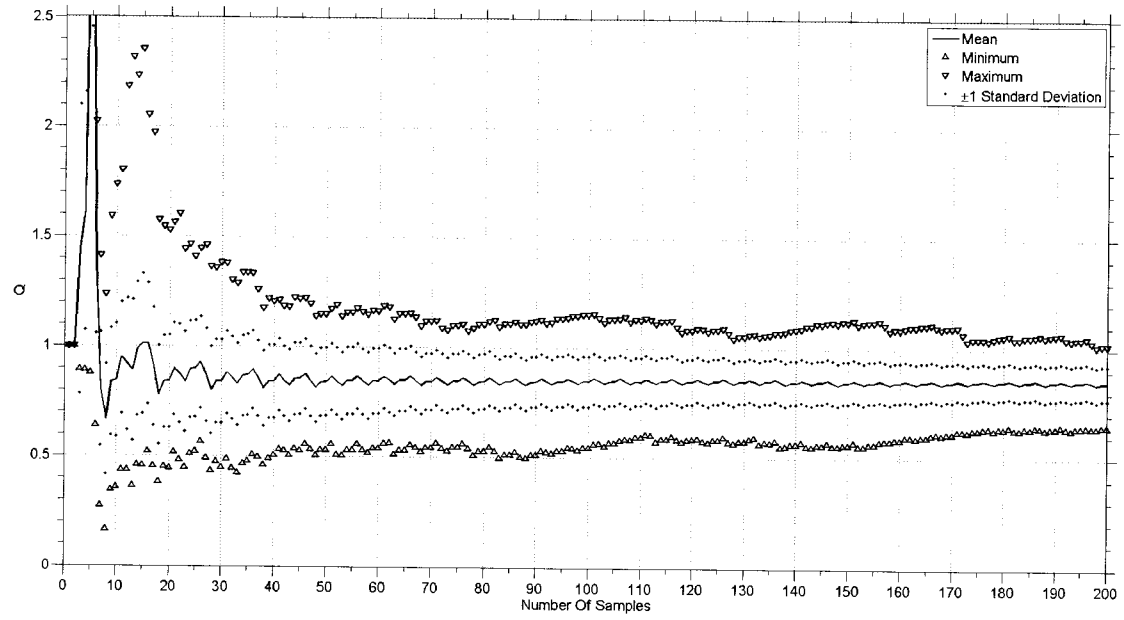


Figure 4.15: Variation in alias suppression effectiveness of jitter sampling with 5% Gaussian jitter. Statistical properties for 100 trials are shown.

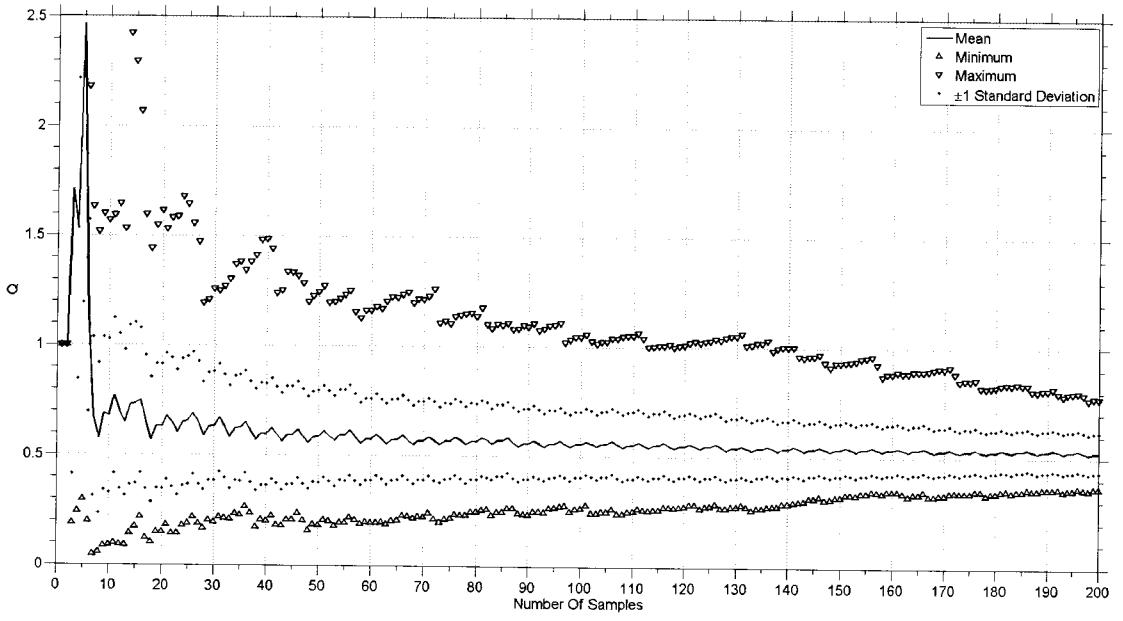


Figure 4.16: Variation in alias suppression effectiveness of jitter sampling with 10% Gaussian jitter. Statistical properties for 100 trials are shown.

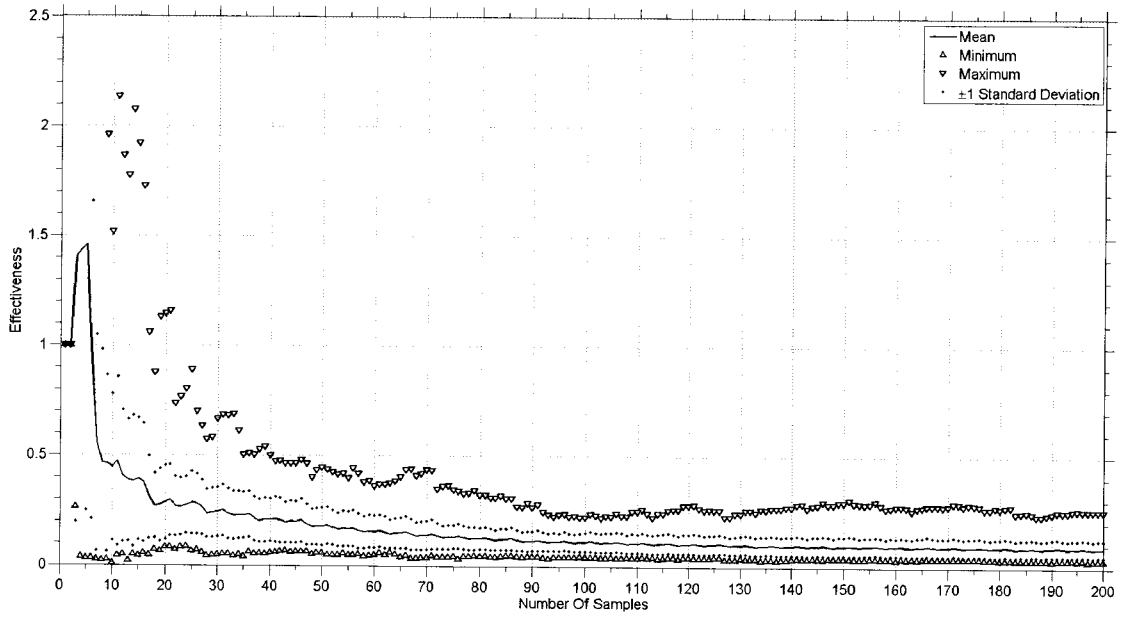


Figure 4.17: Variation in alias suppression effectiveness of jitter sampling with 20% Gaussian jitter. Statistical properties for 100 trials are shown.

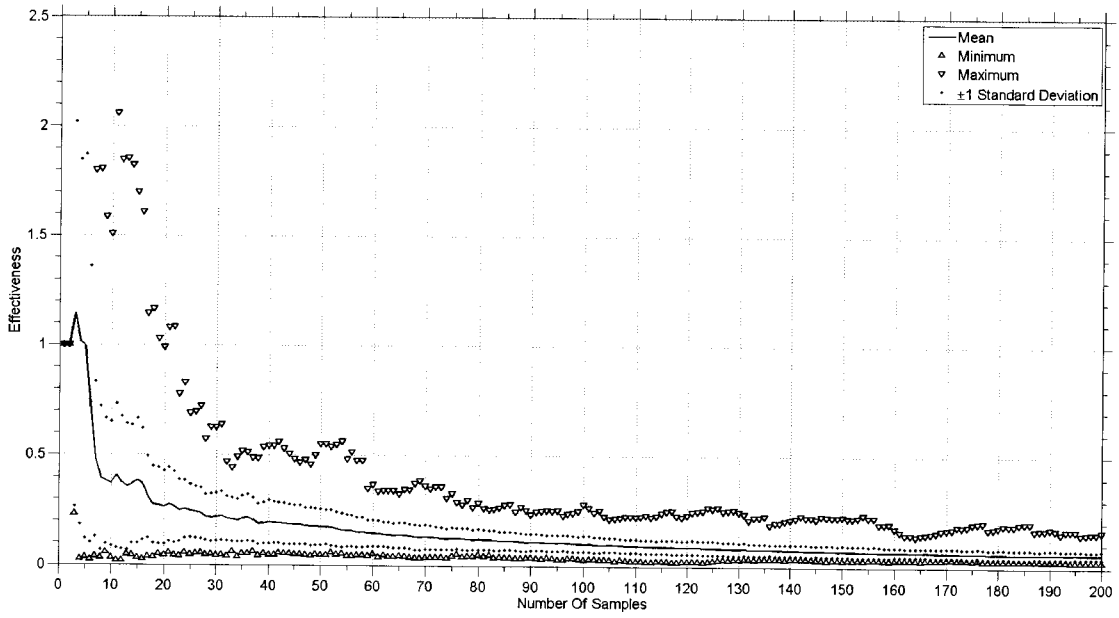


Figure 4.18: Variation in alias suppression effectiveness of jitter sampling with 30% Gaussian jitter. Statistical properties for 100 trials are shown.

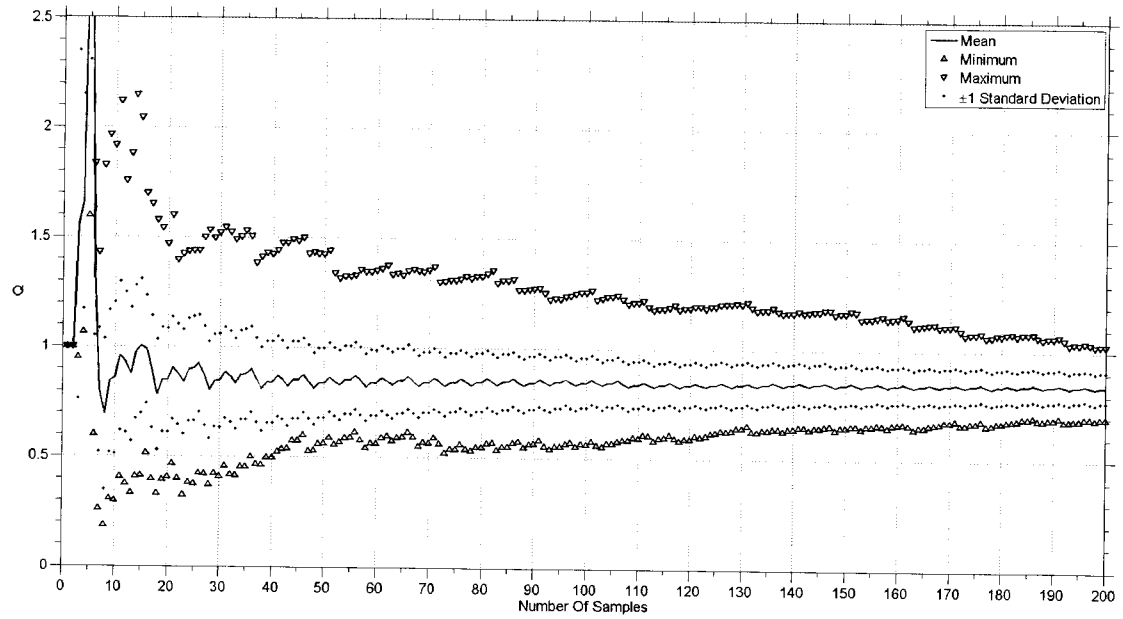


Figure 4.19: Variation in alias suppression effectiveness of jitter sampling with 5% uniform jitter. Statistical properties for 100 trials are shown.

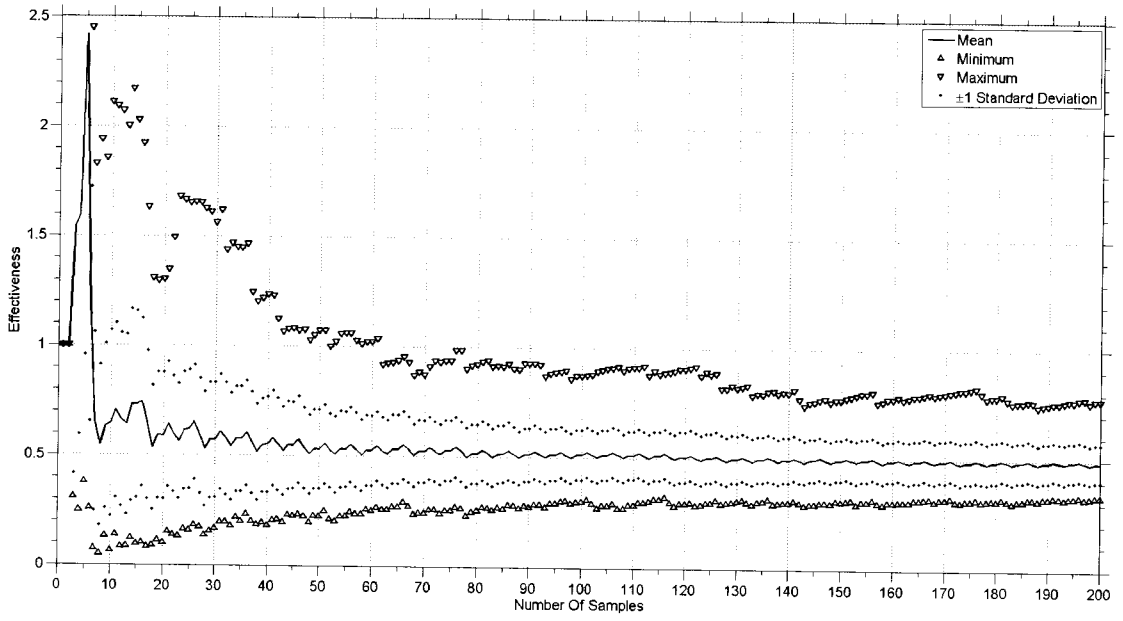


Figure 4.20: Variation in alias suppression effectiveness of jitter sampling with 10% uniform jitter. Statistical properties for 100 trials are shown.

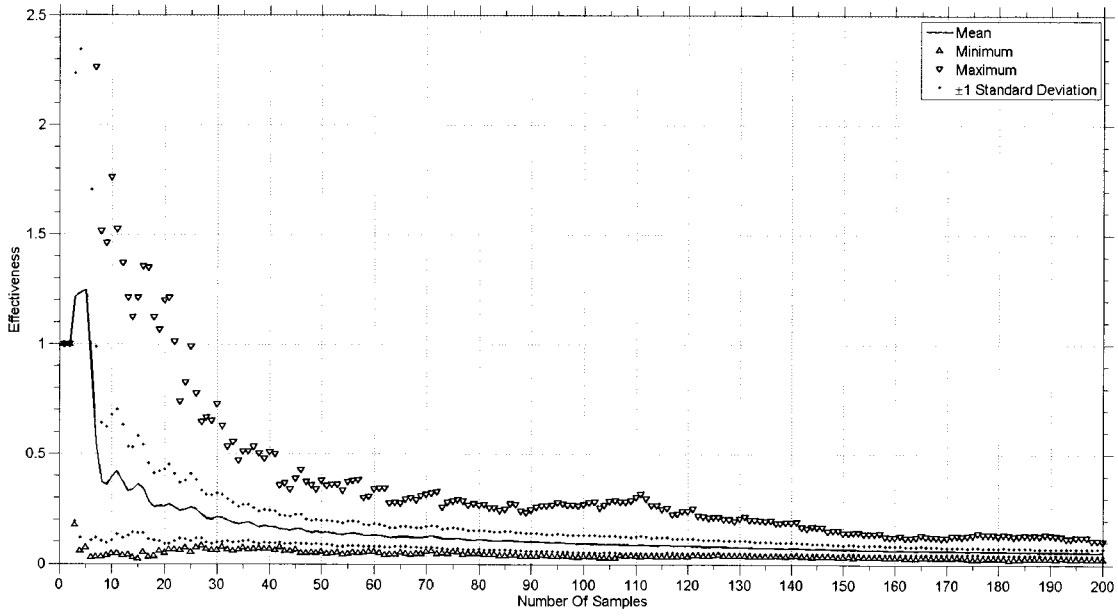


Figure 4.21: Variation in alias suppression effectiveness of jitter sampling with 20% uniform jitter. Statistical properties for 100 trials are shown.

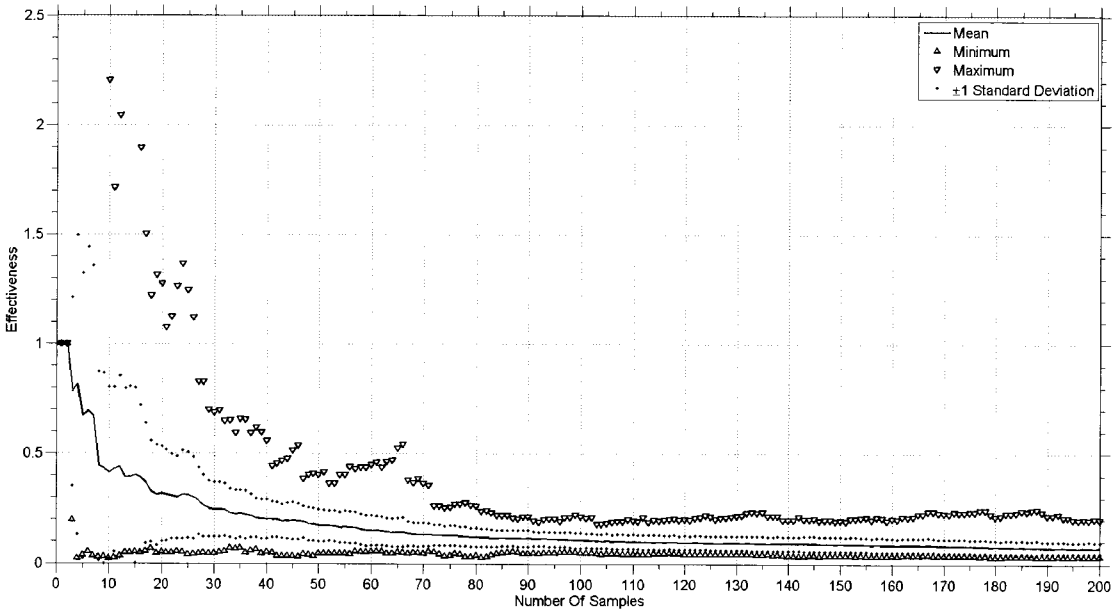


Figure 4.22: Variation in alias suppression effectiveness of jitter sampling with 30% uniform jitter. Statistical properties for 100 trials are shown.

Figure 4.15 through Figure 4.22 repeat the conditions from Figure 4.11 - Figure 4.14, but add information about the variation encountered. As can be seen, chance variation is a factor in the outcome: the best case is extremely good, while the worst case intolerably bad.

4.5 "Deterministic Jitter"

The variation in results shown in the previous section (Section 4.4) is sufficiently large that benefit could be obtained by pre-selecting the jitter values to guarantee the best-case outcomes. At the very least, constraining the generated jitter values to bound the worst to be no worse than the no-jitter case would at least ensure that jitter sampling would not have a detrimental effect.

Various attempts were made to identify patterns in the jitter of the high-performing cases that would distinguish them in advance from the detrimental cases, but no consistent properties could be identified. A subsequent effort was made to use the Farey sequence as the basis for a sampling sequence, on the basis of belief that its construction would have the necessary properties. The results of this effort are discussed in the next section.

4.5.1 Farey Sequence

The Farey sequence F_n is the ordered set of fractions between 0 and 1 whose denominator is less than or equal to n.[1] The first five Farey Sequences are

[1]The sequence is always expressed in a strictly increasing order, and it is implicit in the definition that the fractions are always considered in simplest terms.

$$
\begin{aligned}
F_1 &= \{0,1\} \\
F_2 &= \left\{0,\frac{1}{2},1\right\} \\
F_3 &= \left\{0,\frac{1}{3},\frac{1}{2},\frac{2}{3},1\right\} \\
F_4 &= \left\{0,\frac{1}{4},\frac{1}{3},\frac{1}{2},\frac{2}{3},\frac{3}{4},1\right\} \\
F_5 &= \left\{0,\frac{1}{5},\frac{1}{4},\frac{1}{3},\frac{2}{5},\frac{1}{2},\frac{3}{5},\frac{2}{3},\frac{3}{4},\frac{4}{5},1\right\}.
\end{aligned}
$$

These are depicted in timeline form, and continued to $F_n|n = 18$ in Figure 4.23.

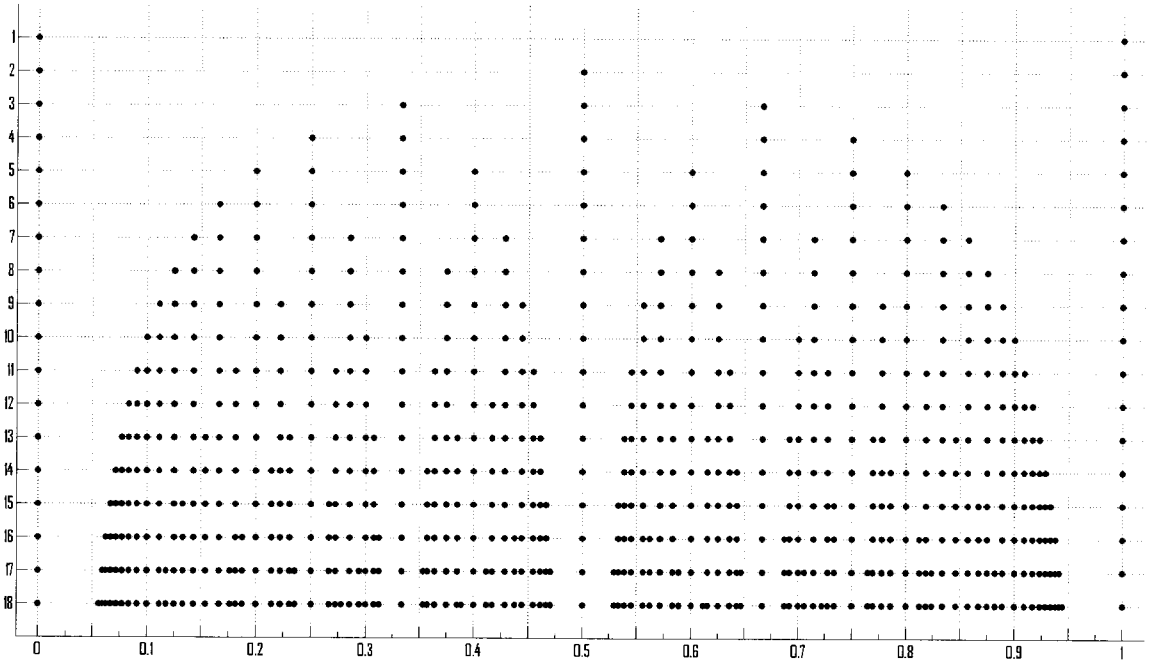


Figure 4.23: Graphical depiction of the first 18 iterations of the Farey Sequence.

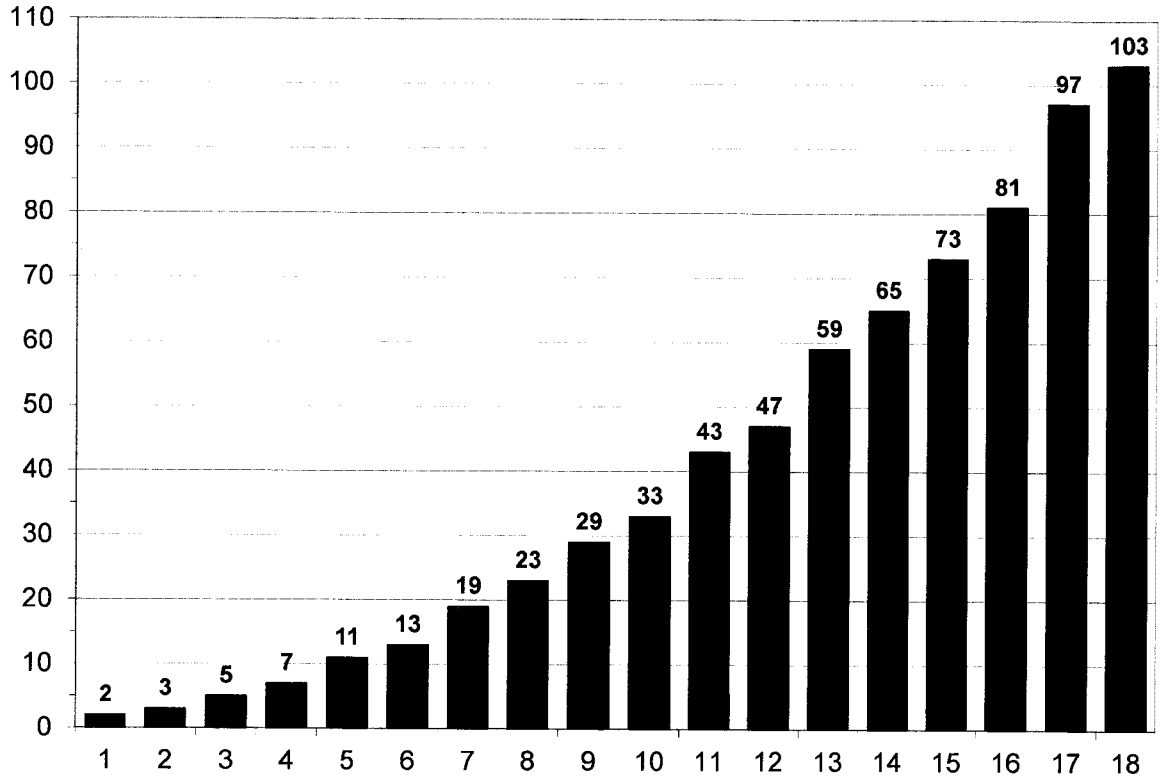


Figure 4.24: The number of timepoints in each iteration of the Farey Sequence.

The Farey sequences were converted to time sequences such that each average interval between successive points in a given time sequence was equal to 1.0 s. These time sequences were then applied as sampling schemes to a variant of the synthetic data model.

The variant data model used in this case, like the standard synthetic data model, follows Equation 4.2, but set $A_1 = 1$, $\omega_1 = 1 \Rightarrow f_1 = 0.159$, $\phi_1 = 0$, $A_2 = 0.5$, $\omega_2 = 10 \Rightarrow f_2 = 1.59$ and $\phi_1 = 0$, giving

$$y(t) = \cos(t) + \frac{1}{2}\cos(10t). \tag{4.4}$$

Note that for this signal, the expected Q value (Section 4.3.3) with no alias suppression is $Q = 0.25$.

The resulting spectra are shown in Figure 4.25.

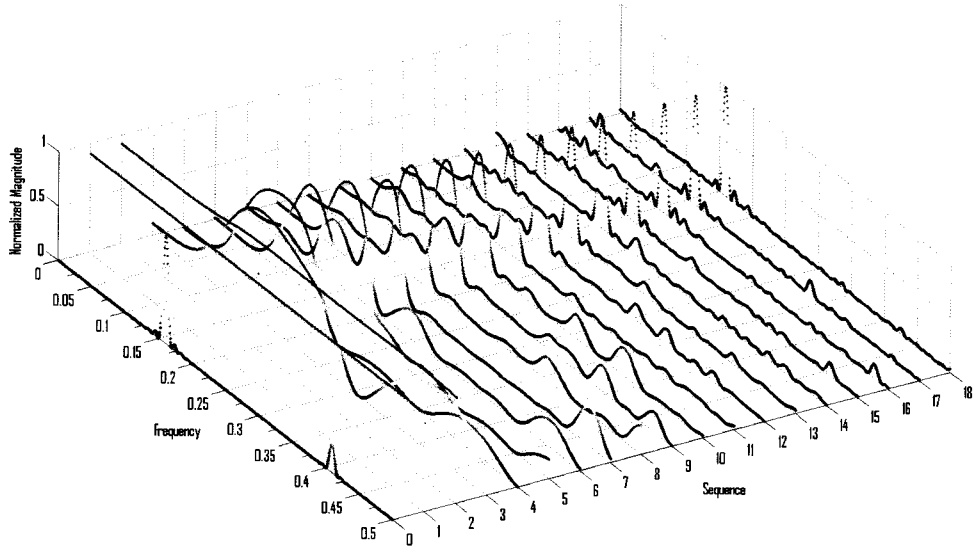


Figure 4.25: A three-dimensional plot of spectra from successive Farey sequences. The sequence labelled "0" is the reference case using unmodified periodic sampling.

The alias suppression effectiveness (Q) calculated from these spectra are shown in Figure 4.26.

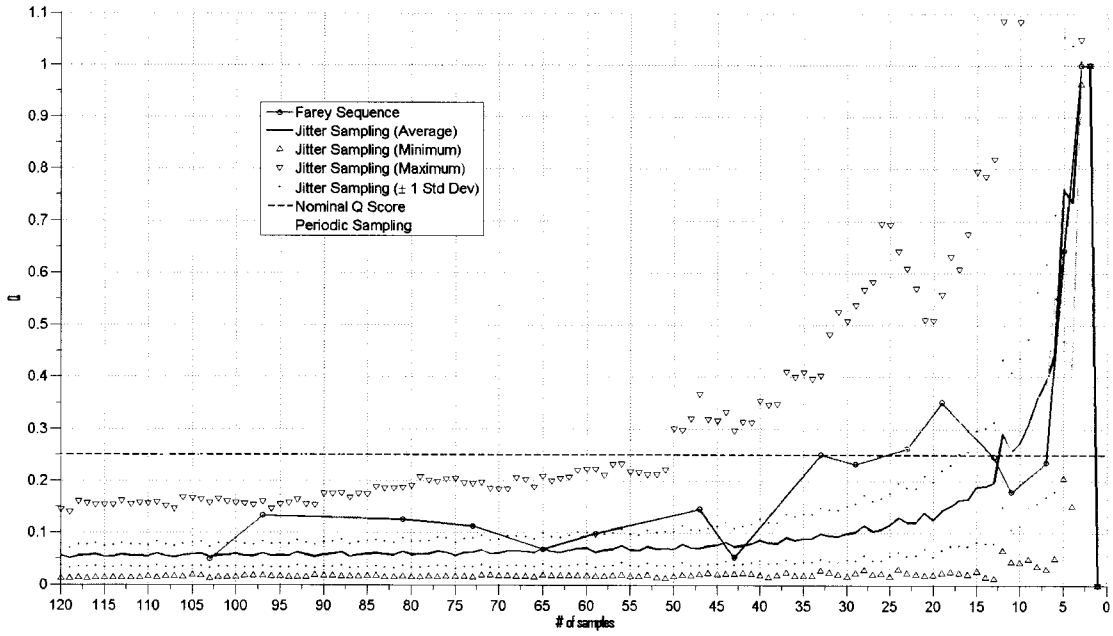


Figure 4.26: Alias suppression effectiveness of a Farey Sequence based sampling regime compared with with jitter sampling using 10% Gaussian jitter and non-jittered periodic sampling. Note that the reference Q ratio for this trial is 0.25, which is also marked.

While the Farey sequence sampling does sometimes outperform jitter sampling, this is not consistent. Furthermore, it even underperforms relative to periodic sampling in some instances. Accordingly, Farey sequence based sampling cannot be recommended.

4.6 Demonstration In GalSim

Figure 4.27 through Figure 4.34 show the results of applying jitter sampling to the GalSim model (compare Section 3.5, especially Figure 3.5, for the non-jitter sampling results).

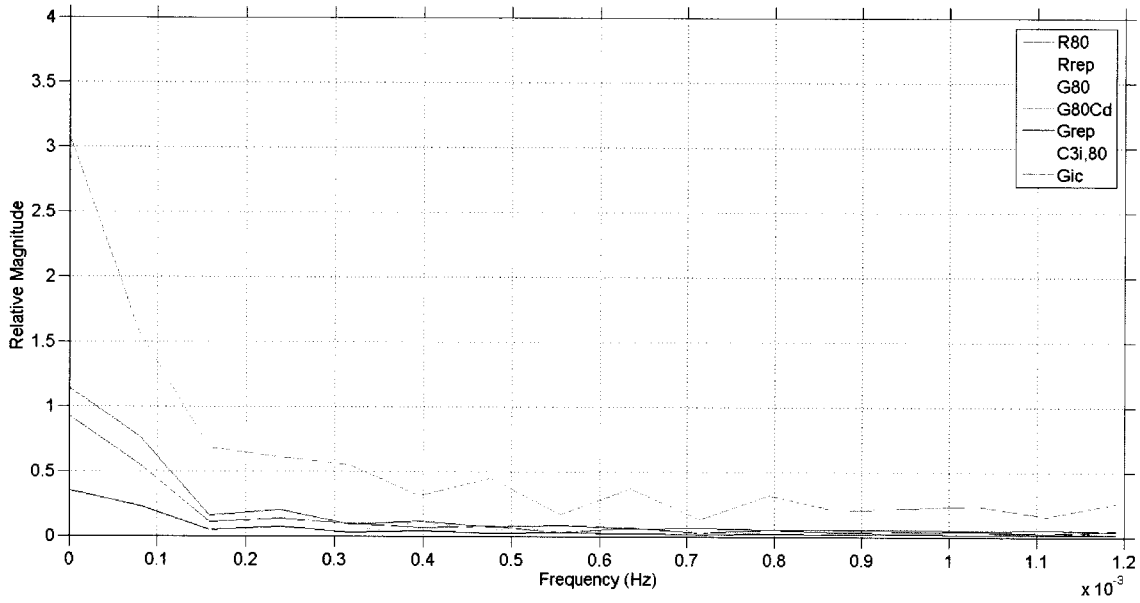


Figure 4.27: Spectrum of select GalSim species with 10% Gaussian jitter sampling. Results shown are the average of 100 trials and values are normalized as per Section 3.4.1.

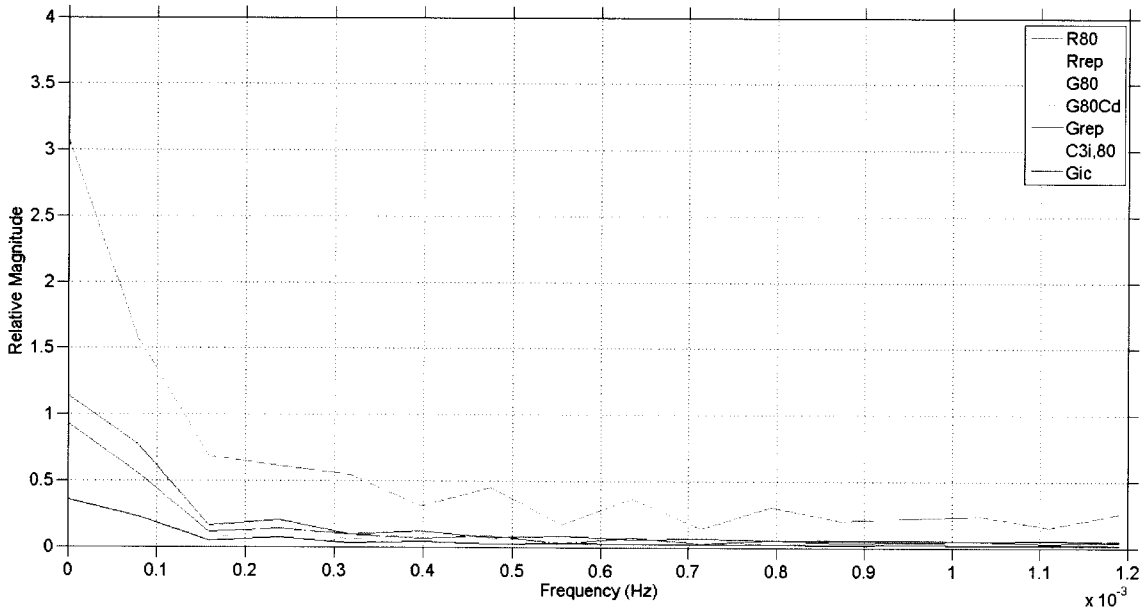


Figure 4.28: Spectrum of select GalSim species with 20% Gaussian jitter sampling. Results shown are the average of 100 trials and values are normalized as per Section 3.4.1.

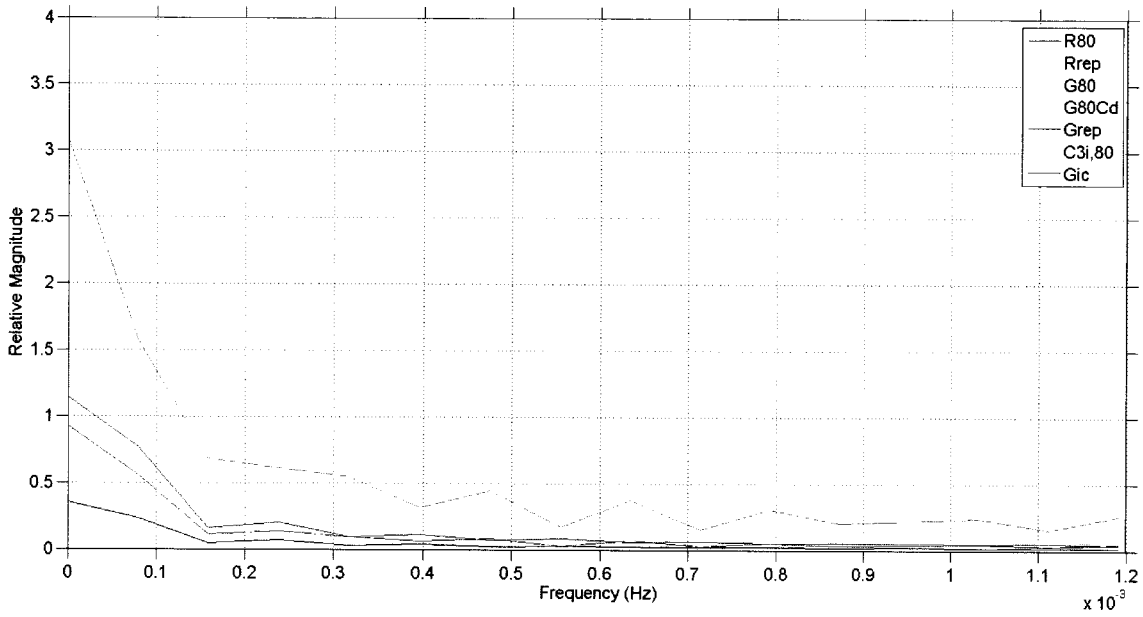


Figure 4.29: Spectrum of select GalSim species with 30% Gaussian jitter sampling. Results shown are the average of 100 trials and values are normalized as per Section 3.4.1.

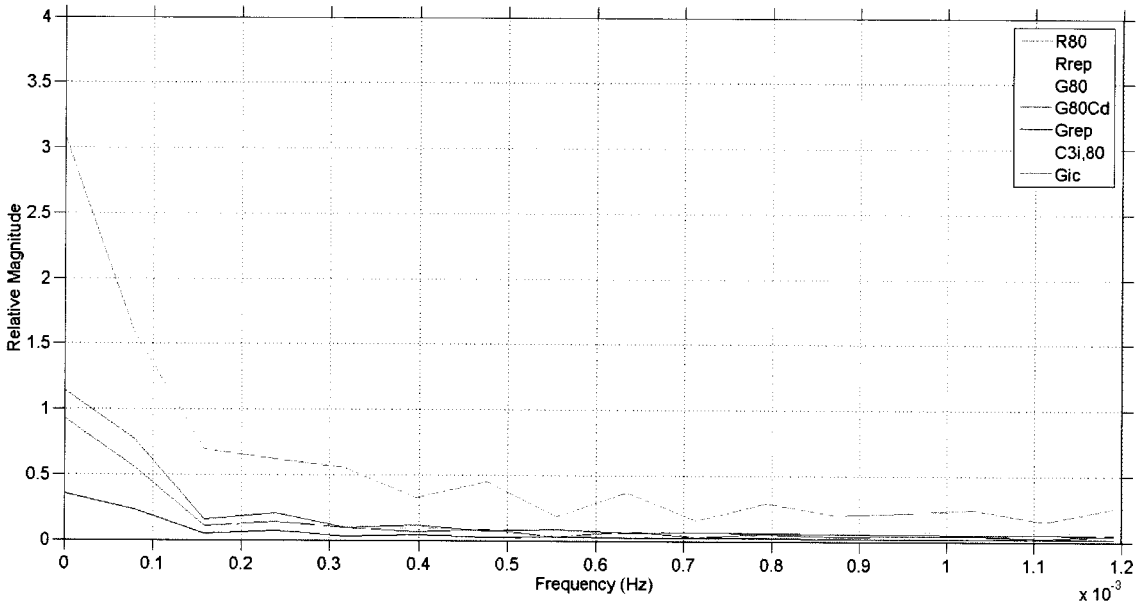


Figure 4.30: Spectrum of select GalSim species with 50% Gaussian jitter sampling. Results shown are the average of 100 trials and values are normalized as per Section 3.4.1.

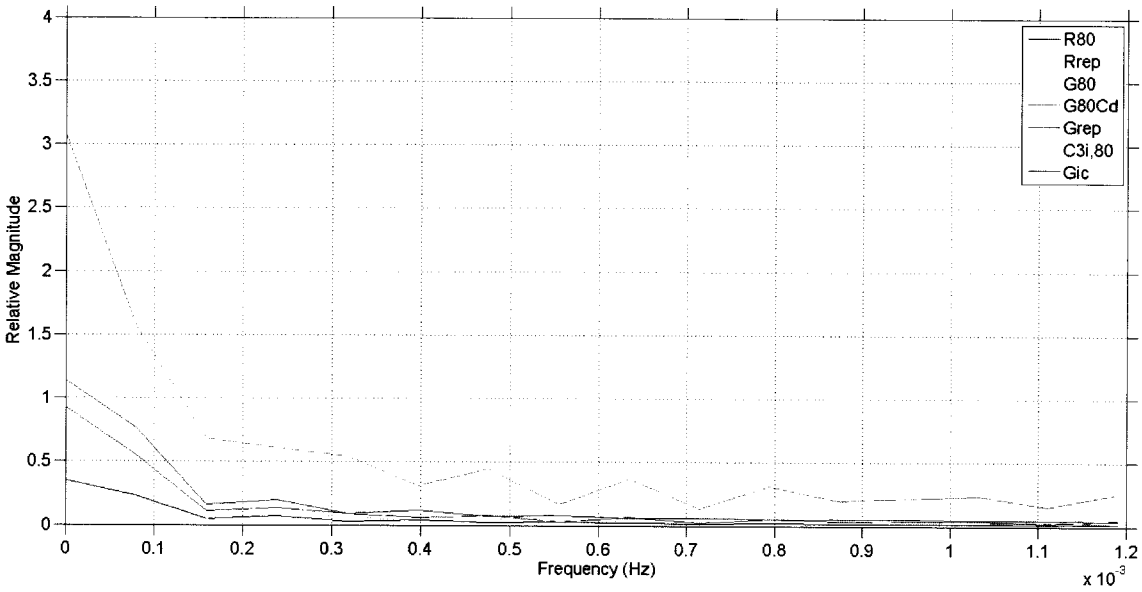


Figure 4.31: Spectrum of select GalSim species with 10% uniform jitter sampling. Results shown are the average of 100 trials and values are normalized as per Section 3.4.1.

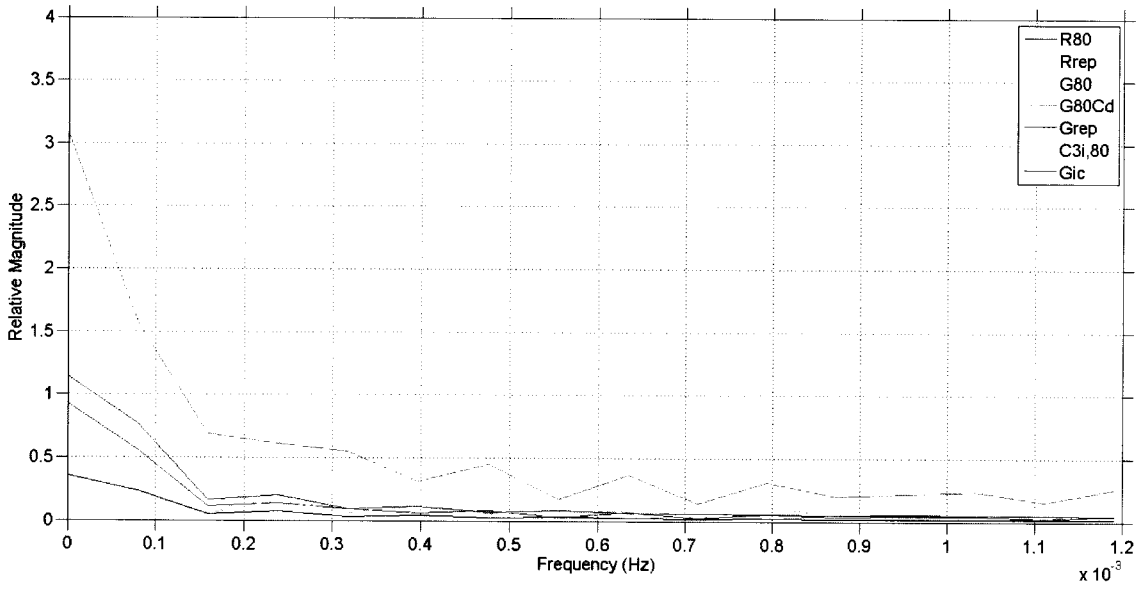


Figure 4.32: Spectrum of select GalSim species with 20% uniform jitter sampling. Results shown are the average of 100 trials and values are normalized as per Section 3.4.1.

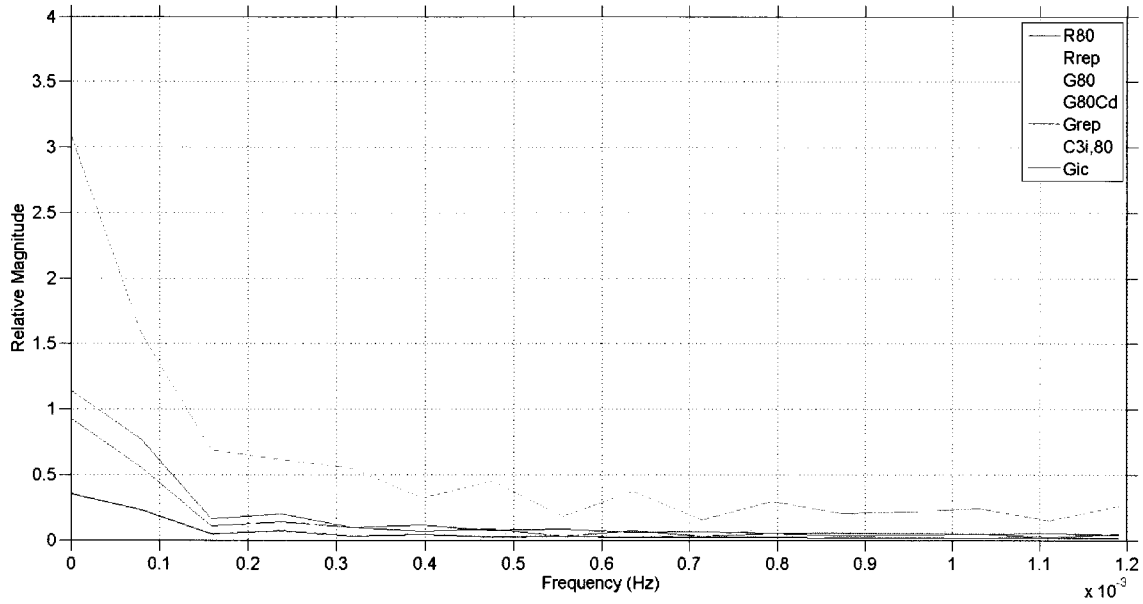


Figure 4.33: Spectrum of select GalSim species with 30% uniform jitter sampling. Results shown are the average of 100 trials and values are normalized as per Section 3.4.1.

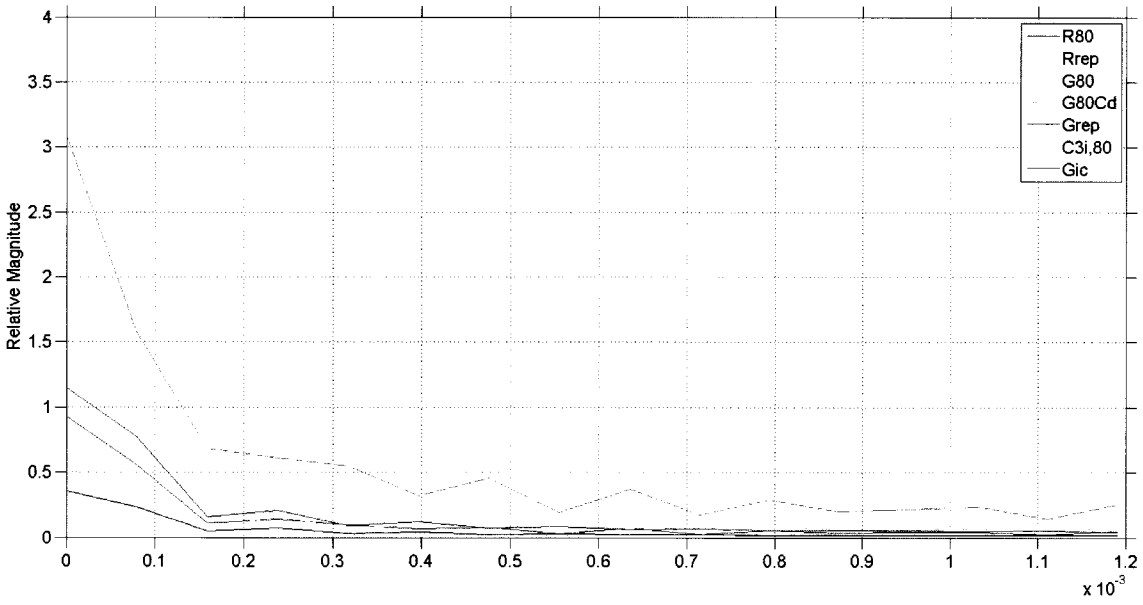


Figure 4.34: Spectrum of select GalSim species with 50% uniform jitter sampling. Results shown are the average of 100 trials and values are normalized as per Section 3.4.1.

Since it is difficult to directly compare these figures to each other, two plots, each showing all cases for a single species, have been produced. In both plots, a no-jitter reference spectrum is also included. Figure 4.35 shows the $C_{3i;80}$ curves (which is subject to the greatest degree of aliasing and therefore shows the most visible effect) from each of the GalSim results above (as well as the no jitter case) plotted on the same set of axes. Similarly, Figure 4.36 shows G_{80} (which is not subject to as much aliasing, and expected to show little effect from jitter sampling) in the same fashion.

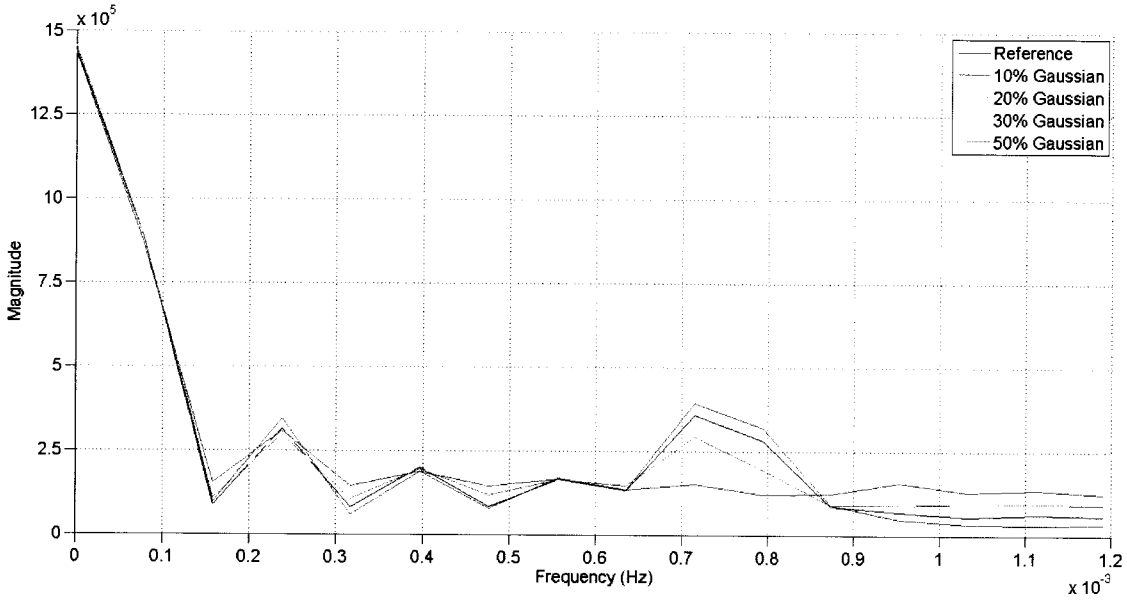


Figure 4.35: Comparison of $C_{3i;80}$ spectrum under various amounts of jitter sampling. "Reference" refers to the no-jitter case.

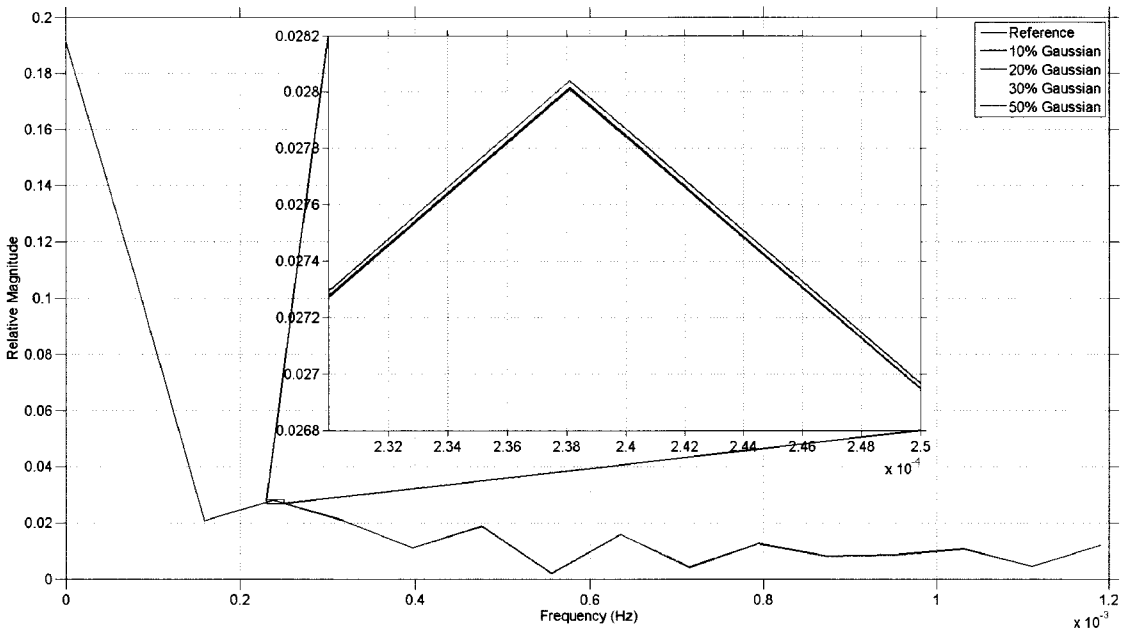


Figure 4.36: Comparison of G_{80} spectrum under various amounts of jitter sampling. "Reference" refers to the no-jitter case.

The results are much as expected. At 7.1×10^{-4} Hz (the point of greatest effect), the $C_{3i;80}$ magnitude varies from 3.916×10^5 in the reference case to 1.403×10^5 in the 50% uniform jitter case. Some (slight) increase in signal intensity is found at other frequencies; this is most likely related to the effects of jitter sampling on spectral leakage, as discussed in Section 4.3.2. Little change is found for G_{80}: all cases lie between 2.800×10^{-2} and 2.804×10^{-2}.

While it is not possible to definitively prove that the decrease in intensity is due to the suppression of aliasing (as opposed to a general attenuation of the signal) several factors strongly suggest that this is the case:

- Jitter sampling has been shown to suppress aliasing (Section 4.3.2);
- Aliasing is known to be occuring in this GalSim scenario (Section 3.5);
- Signal intensity decreases[2] with increasing jitter (consistent with the synthetic

[2]When significant variation is present.

model: Section 4.3.5); and,

- A noticeable decrease in signal intensity is seen with $C_{3i;80}$ (which is aliased), whereas the minimally aliased G_{80} is almost unaffected (Figure 4.35, Figure 4.36).

It is therefore reasonable to conclude that jitter sampling is suppressing aliasing in GalSim.

4.7 Conclusions

Jitter sampling is a well-established technique for reducing aliasing in radio frequency applications; here, it has been shown that it can be applied to TSGES and suppress aliasing when used with biological systems at low sample numbers.

Chapter 5

Time Aggregation and Skip Sampling

TASS is a technique in which the sampling is defined by two elements: the normal set of sampling time points, and a point spread function. This technique was developped independently in this work, but has been studied previously. The most prominent prior use seems to be in the field of econometrics[1]; see for example [35] and [36].

For this implementation, the point spread function defines a distribution of points around the main sampling point. Samples are taken at all of the points in this cluster, and these samples are then physically mixed to average the RNA concentrations. A microarray is then used to measure the gene expression levels of this physically averaged mixture.

5.1 Simple Depiction

The concept of TASS is best illustrated with a simple example. Figure 5.1 shows a typical point spread function in use, with a cluster of three points spaced 0.1 time units apart and centered on the main point.

[1]The study of the statistical properties of economic systems.

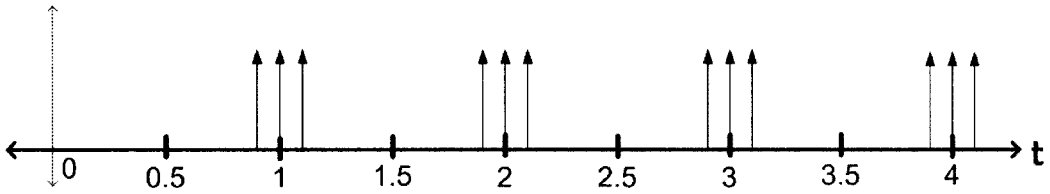


Figure 5.1: An example of the application of a point spread function. Four measurements are made, at $t = 1$, $t = 2$, $t = 3$ and $t = 4$, with three samples each, drawn at -10%, 0% and $+10\%$ of the nominal sampling interval.

5.2 Alternative Interpretation

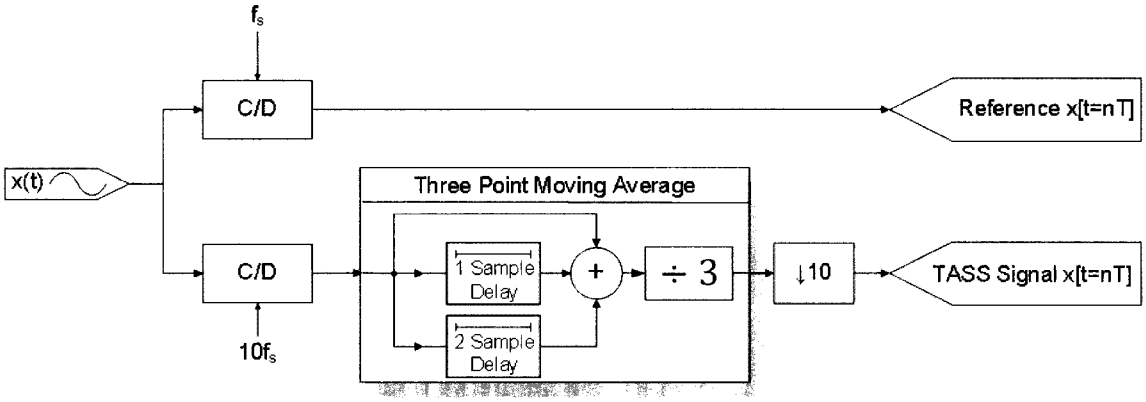


Figure 5.2: Schematic representation of TASS. This is a conceptual representation of one possible implementation.

This approach can also be interpreted in a different way. TASS can be modelled as a combination of oversampling, a time-moving average and downsampling. A block diagram representation for this approach (using the three-element point spread function with samples drawn at ± 0.1 and 0) is shown in Figure 5.2.

5.3 Biology Motivation

The genesis for this technique was speculation regarding the effect of non-idealities in the lab procedure for TSGES. Specifically, the abstracted/ideal view of TSGES is

that of periodic evaluation of the level of gene expression in a single cell; the reality is that the testing procedure requires a large number of similar cells (typically on the order of 10,000) and it necessarily involves the destruction of the cells [37].

The large number of cells in each sample, combined with the fact that cells are not perfectly in sync [38], implies that the resulting sample measurements will be an average of many closely-spaced, but slightly different values. TASS is essentially an attempt to refine this behaviour in a more deterministic and effective fashion.

5.4 Applicability to TSGESs

The normal procedure for TSGES consists of[2]:

1. At each time-point:
   (a) Collect a sample.
   (b) Preserve the sample.
   (c) Apply the sample to a microarray.
2. Collect the results from all microarrays.
3. Analyze the collected data.

TASS can be implemented by collecting multiple samples at each time point, and mixing them before applying them to the microarray. The modified procedure is then:

1. At each main time-point:
   (a) For each sub-point in the point spread function:
      i. Collect a sample.

[2]See Section 2.1.3 for a more complete description.

ii. Preserve the sample.

(b) Mix the preserved samples together.

(c) Apply the mixed sample to a microarray.

2. Collect the results from all microarrays.

3. Analyze the collected data.

In practice, the handling of the samples is not time-sensitive after they have been preserved (using the buffer described in Section 2.1.3), and so all of the microarrays are run simultaneously, after the collection phase is complete. In this case, the sample mixing can be similarly deferred.

The collection of many more samples than the number of available microarrays does not appear to be a significant impediment or expense; see [26] discussed in Section 2.3.

5.5 Demonstration In Synthetic Data

5.5.1 Synthetic Data Model

Before applying TASS to the GAL regulon model, it is helpful to test it in a much simpler system with easier to interpret results. The system presented in Section 4.3.1, defined by the linear superposition of two sinusoidal signals, is used here.

5.5.2 Application of TASS

Figure 5.3 shows a time-domain plot of the synthetic data model along with TASS sampling (using a point spread function of $-20\%, -10\%, 0, +10\%, +20\%$ and a nominal sampling rate of 1 Hz) and its results. The frequency domain depiction of

the same results is shown in Figure 5.4.

Comparing Figure 5.4 with Figure 4.4 (note the decrease in magnitude of the aliased 0.3 Hz peak) gives a clear indication that TASS can be effective in suppressing aliasing.

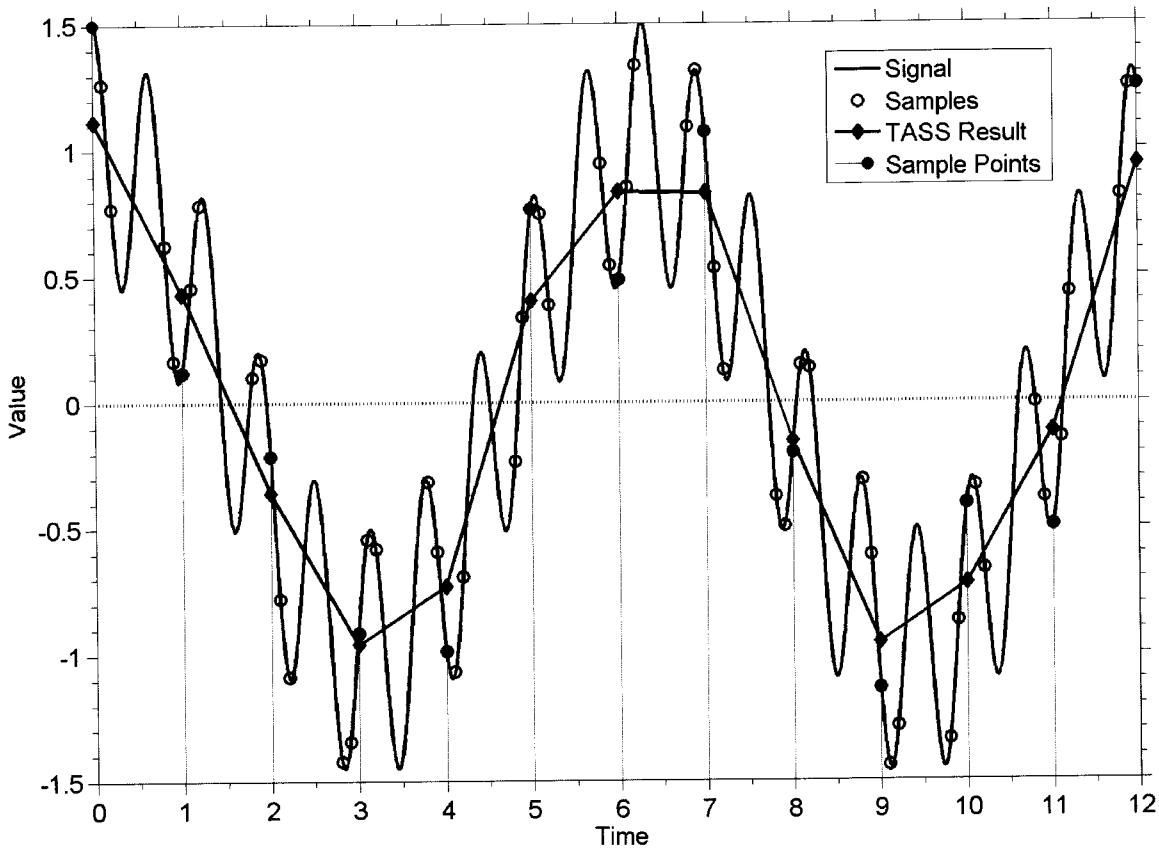


Figure 5.3: Time domain depiction of TASS. The nominal sampling rate is 1 Hz, and the point spread function is $-20\%, -10\%, 0, +10\%, +20\%$.

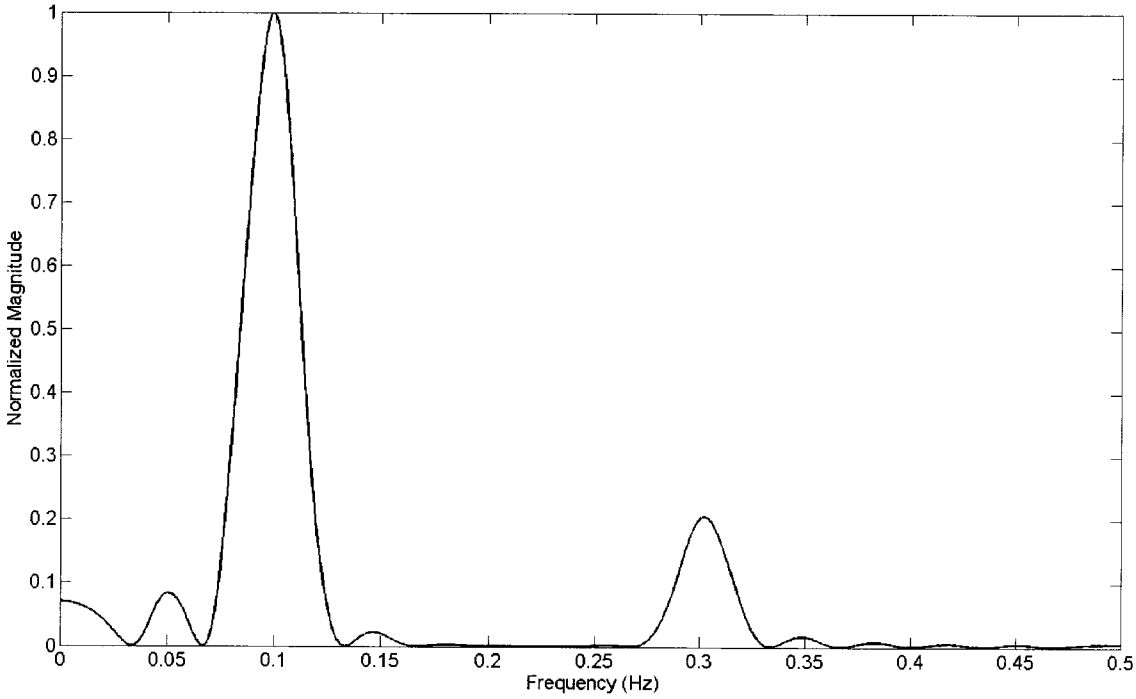


Figure 5.4: Spectrum of the synthetic data model with TASS applied. This is the frequency domain depiction of the TASS function from Figure 5.3. Compare to Figure 4.4.

5.5.3 Measuring Alias Suppression

A simple, useful measure of the alias suppression effectiveness of TASS in the synthetic data model is $Q = \frac{M_1}{M_2}$, the ratio of the magnitudes of the two spectrum peaks. This is the same metric as was used with jitter sampling, and the details of the technique can be found in Section 4.3.3.

5.5.4 Effect of Number of Samples

One of the important considerations in this work is that the number of samples available is at least an order of magnitude fewer than in more conventional DSP applications, and the quality of the signal interpretation declines with decreasing numbers of samples. It is therefore critical to determine how resilient the effect of

TASS is to low number of samples.

Figure 5.5 shows the measured spectrum of the synthetic model at a range of sample numbers. This can be thought of as a "stack" of spectra, each one like the one shown in Figure 5.4, but using a different number of samples. This is done by generating a generating a series of 200 samples taken at one second intervals from the synthetic data model (with the application of TASS), and then calculating the frequency content after truncating the series to the desired length. Evaluating each spectrum as per Section 4.3.3, and plotting number of samples versus Q, we can see that alias suppression improves with increasing sample number, as shown in Figure 5.6, though the effect appears to plateau above approximately fifty samples.[3] As was the case with jitter sampling, results below 10 samples are unreliable due to lack of resolution.

[3]The reason for this plateau is unknown, but is suspected to be due to the fact that TASS, unlike jitter sampling, is a deterministic process.

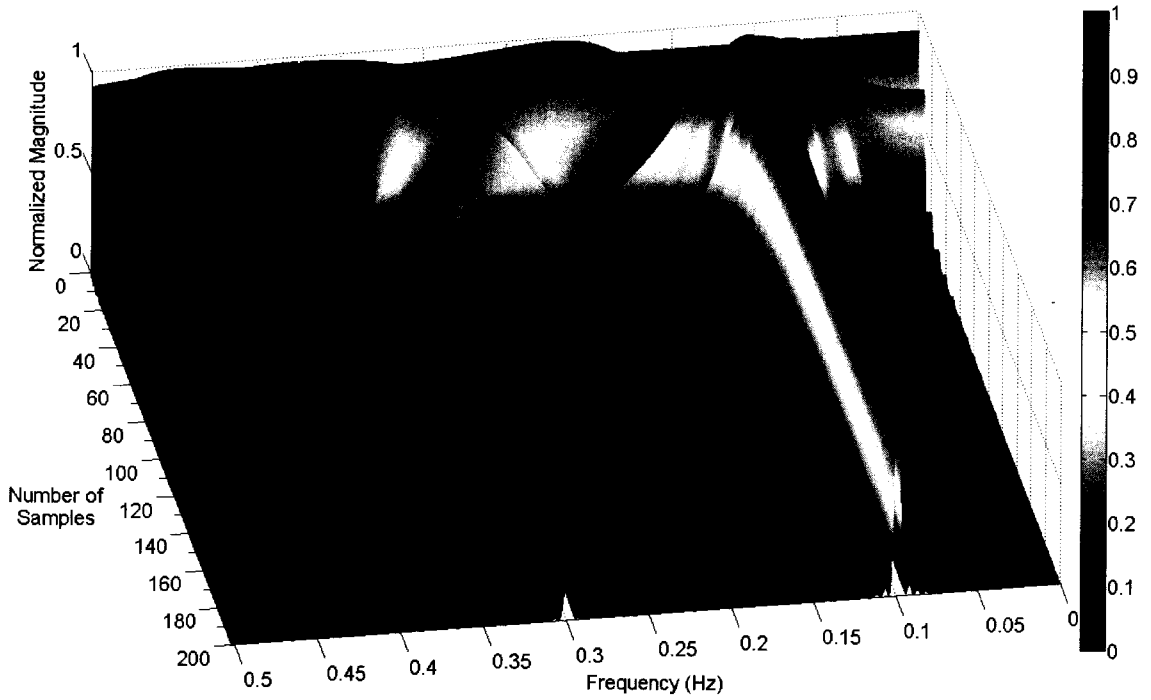


Figure 5.5: A three-dimensional plot of spectra at various sample numbers. TASS was applied with a nominal sampling rate of 1 Hz, and the point spread function is $-20\%, -10\%, 0, +10\%, +20\%$.

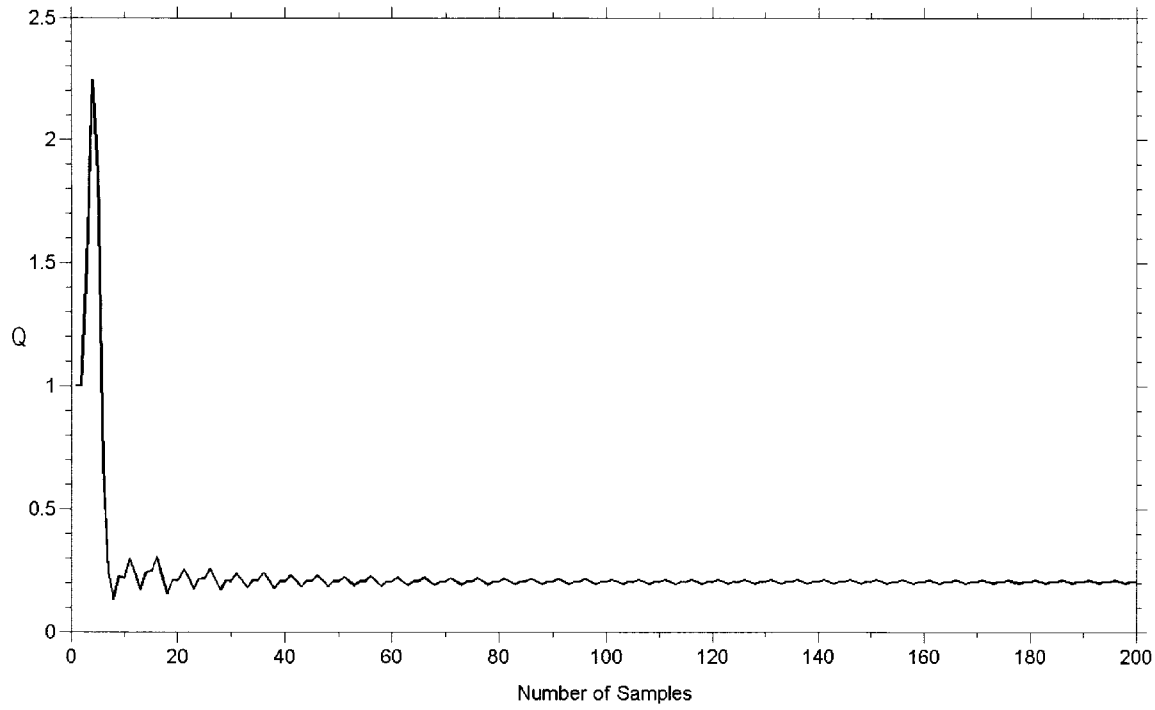


Figure 5.6: Alias suppression effectiveness of TASS versus number of samples. The data shown are the Q values of the spectra shown in Figure 5.5.

5.5.5 Effect of Sample Spread

Obviously, one of the most important considerations in the use of TASS is the selection of the point spread function. A representative sample of the various functions evaluated in the course of this research is shown in Figure 5.7 and the results for these (as applied to the synthetic data model and evaluated using Q) are shown in Figure 5.8 through Figure 5.12. The alias suppression effectiveness (evaluated at the arbitrary choice of 30 samples) varied between $Q = 0.7075$ (for the second case, $-10\%, -5\%, 0\%, +5\%, +10\%$) and $Q = 0.01284$ (for the fifth case, $-30\%, -20\%, -10\%, 0, +10\%, +20\%, +30\%$). Based on these results, it seems that the greater the extent of the spread, the greater the alias suppression effect.

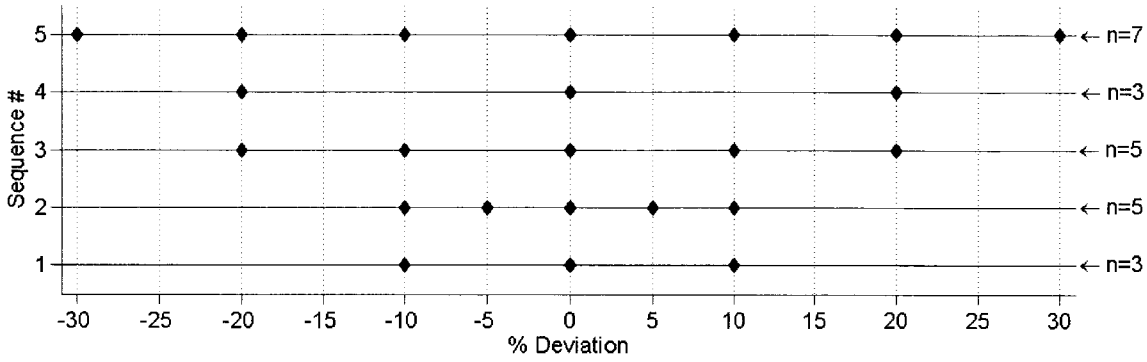


Figure 5.7: A representative sample of TASS point spread functions

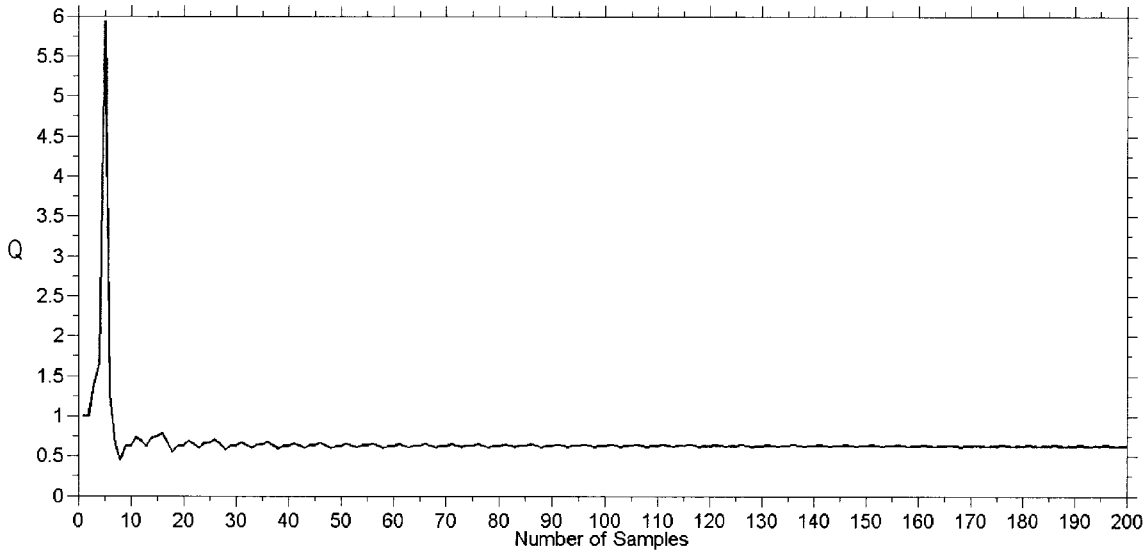


Figure 5.8: Results of the first point spread function from Figure 5.7

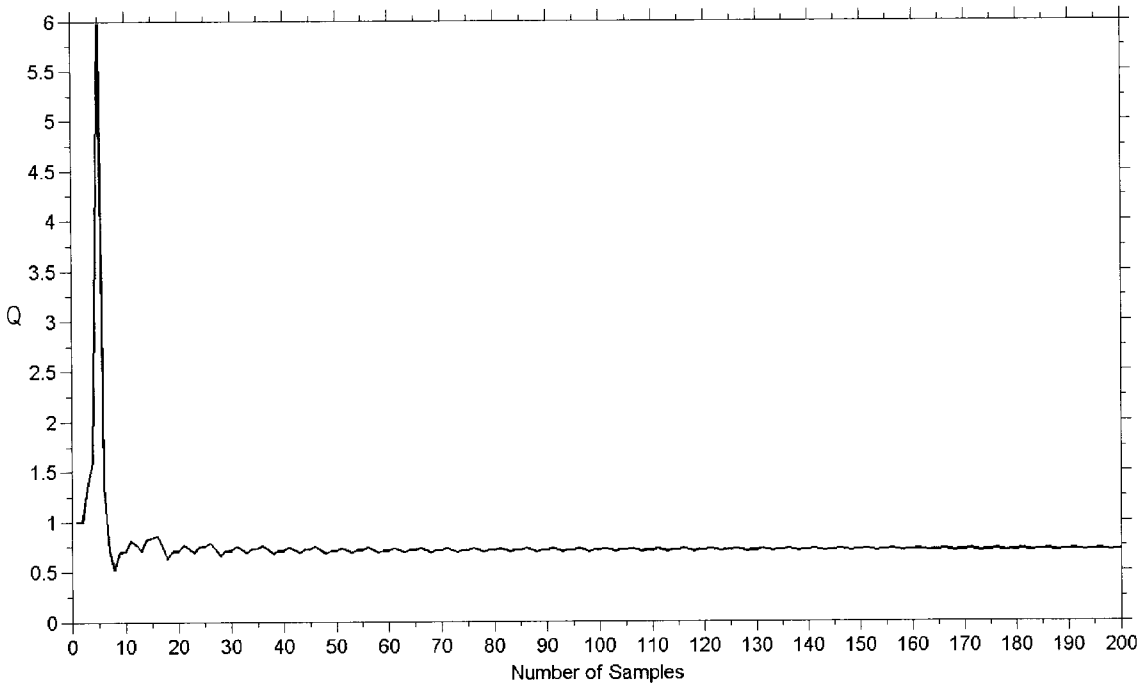


Figure 5.9: Results of the second point spread function from Figure 5.7

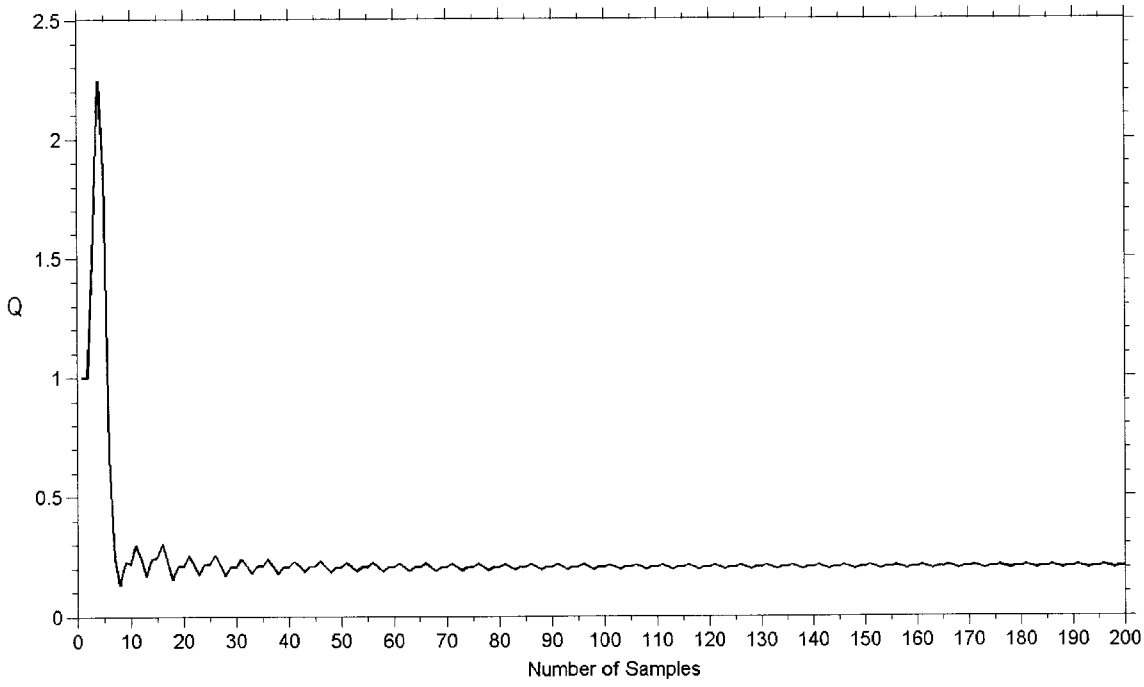


Figure 5.10: Results of the third point spread function from Figure 5.7

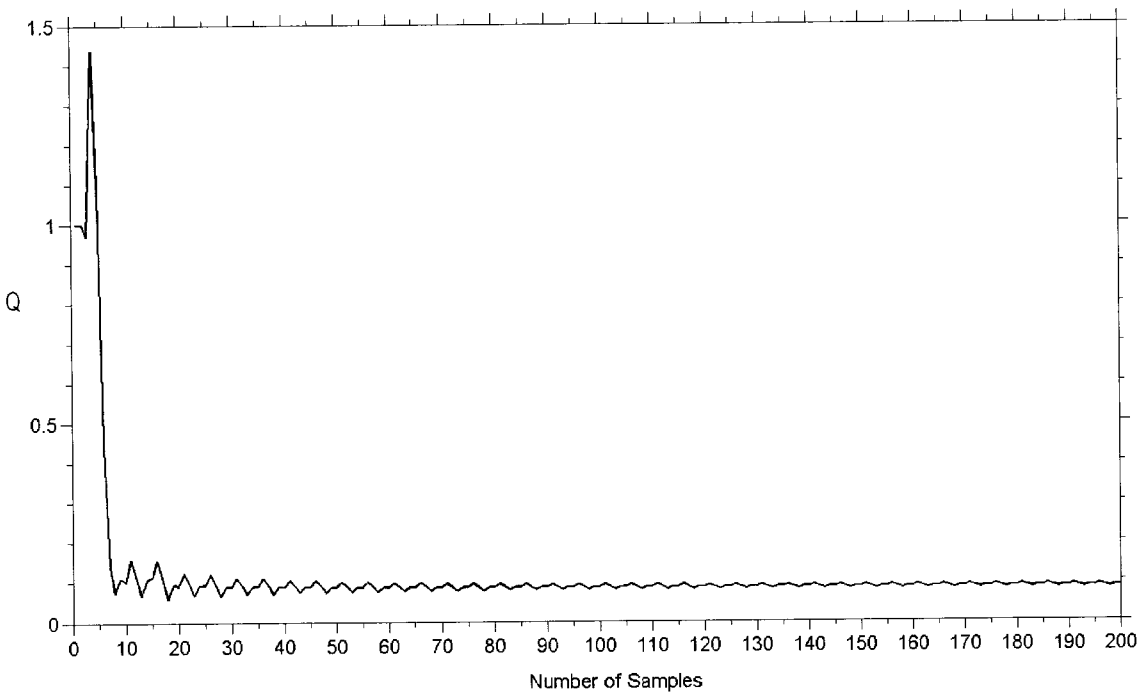


Figure 5.11: Results of the fourth point spread function from Figure 5.7

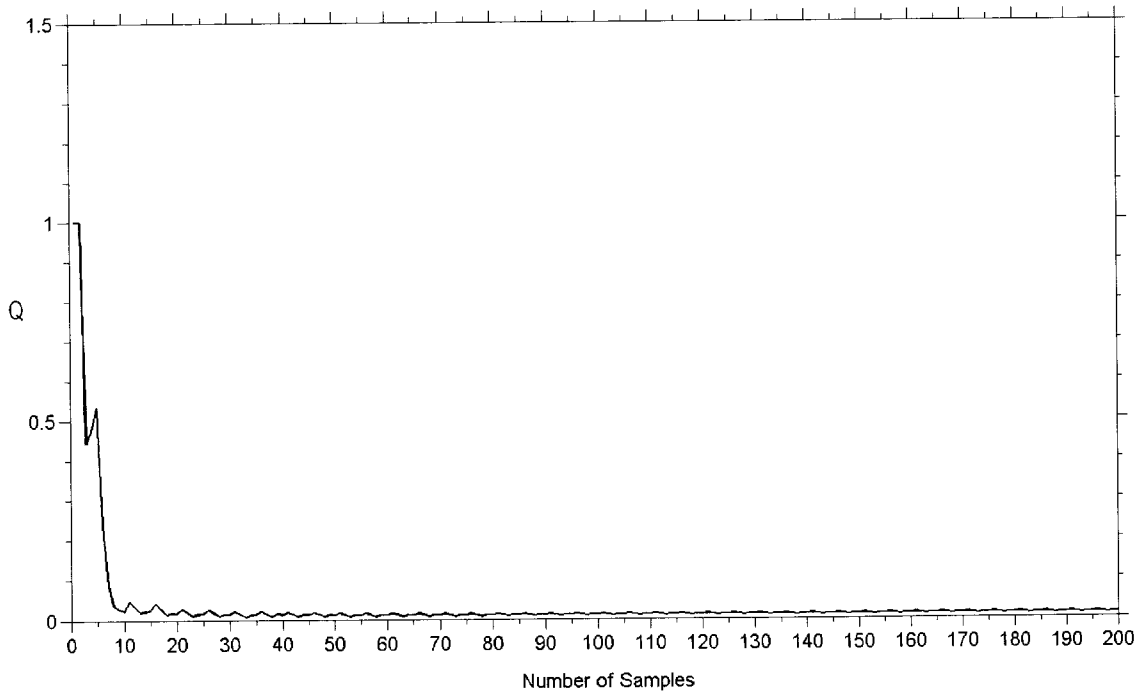


Figure 5.12: Results of the fifth point spread function from Figure 5.7

While a full analysis has yet to be done, it appears that the number of samples and the maximum deviation of the point spread function govern the effectiveness of the technique. This is, however, an important area for future research.

5.6 Demonstration In GalSim

Figure 5.13, Figure 5.14, Figure 5.15, Figure 5.16 and Figure 5.17 show the results of applying TASS to the GalSim model using the point spread functions from Figure 5.7 (compare Section 3.5, especially Figure 3.5, for the non-TASS results).

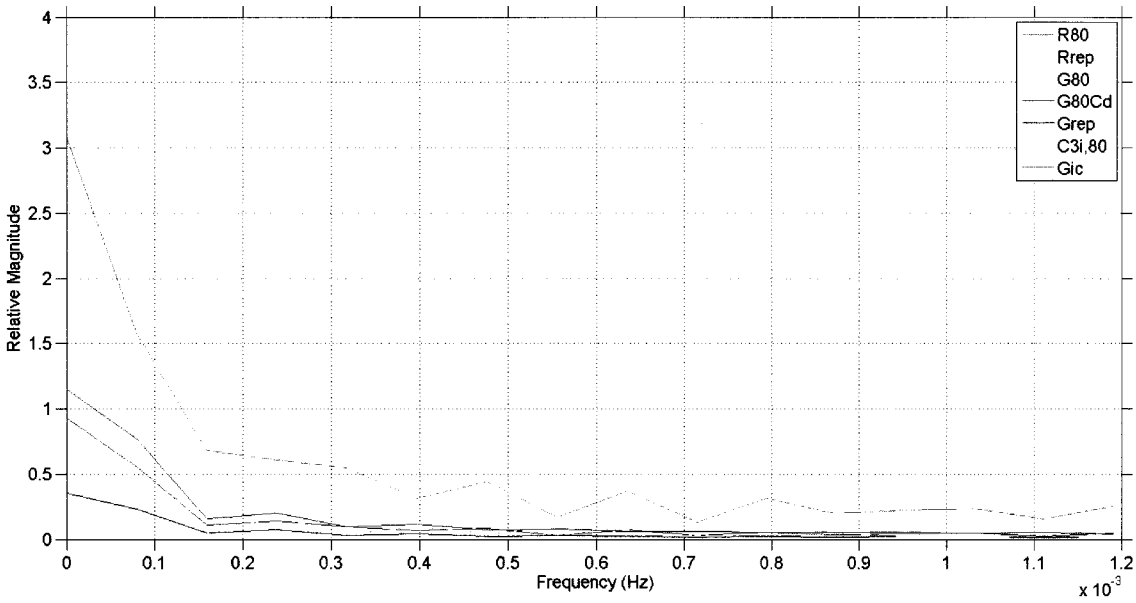


Figure 5.13: Spectrum of select GalSim species with TASS using the point spread function -10%,0,+10%. Values are normalized as per Section 3.4.1.

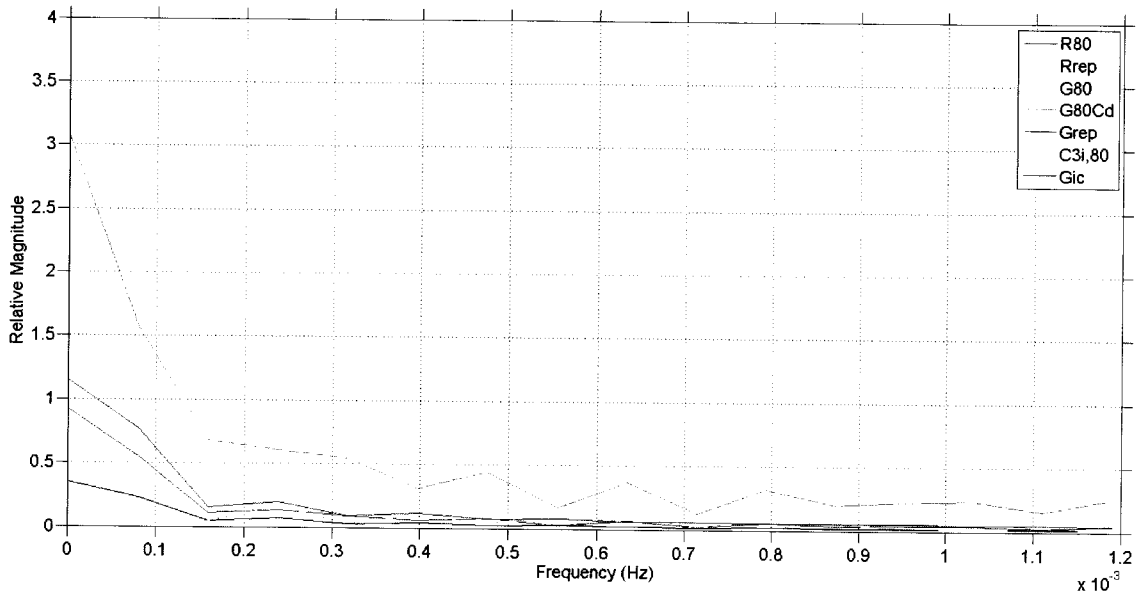


Figure 5.14: Spectrum of select GalSim species with TASS using the point spread function -10%,-5%,0,+5%,+10%. Values are normalized as per Section 3.4.1.

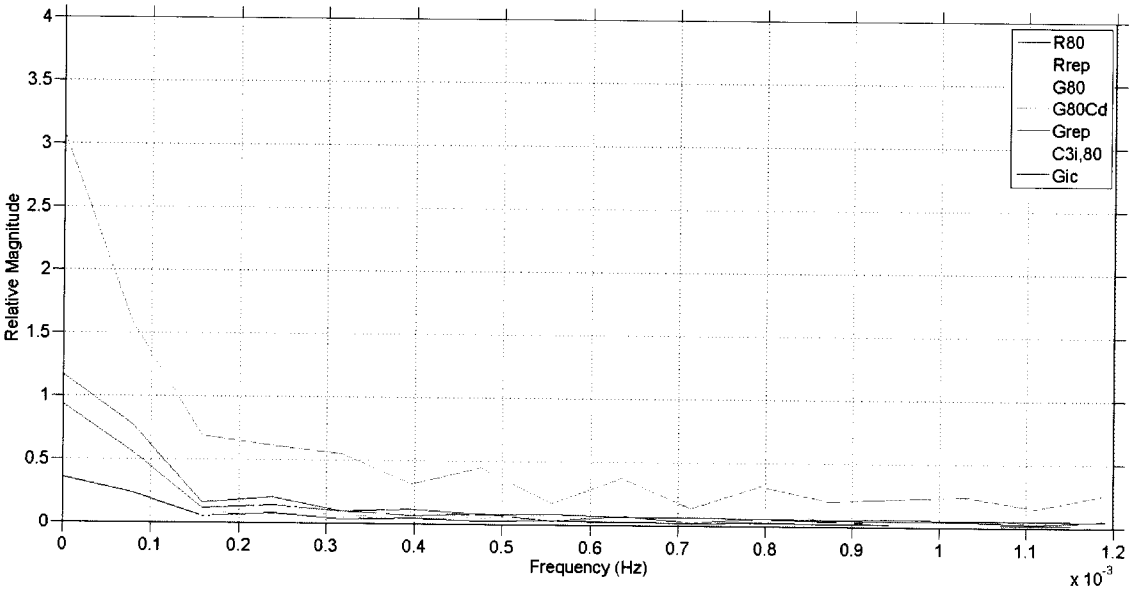


Figure 5.15: Spectrum of select GalSim species with TASS using the point spread function -20%,-10%,0,+10%,+20%. Values are normalized as per Section 3.4.1.

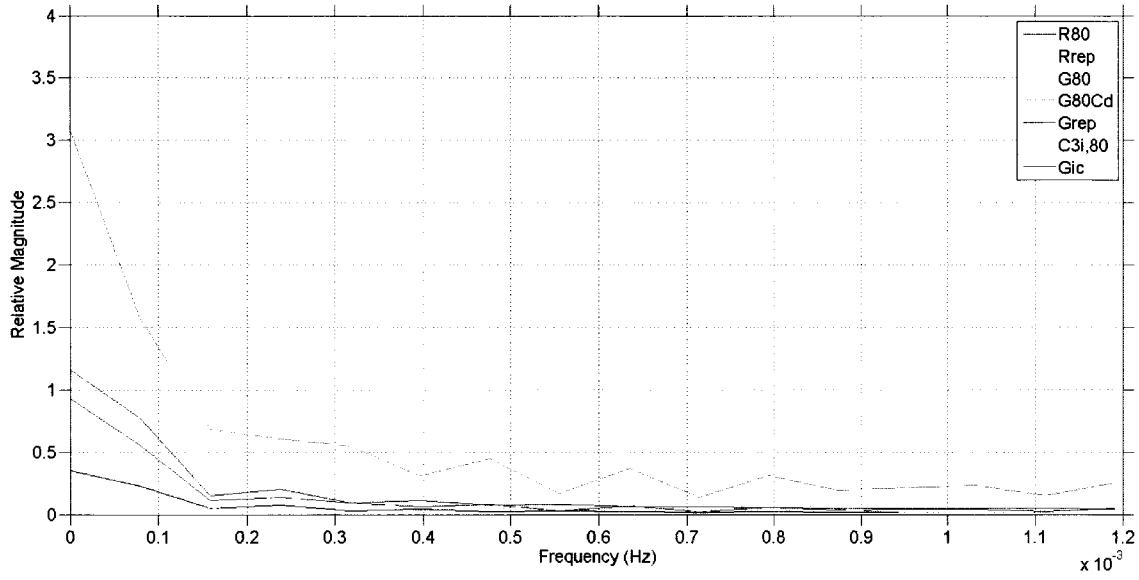


Figure 5.16: Spectrum of select GalSim species with TASS using the point spread function -20%,0,+20%. Values are normalized as per Section 3.4.1.

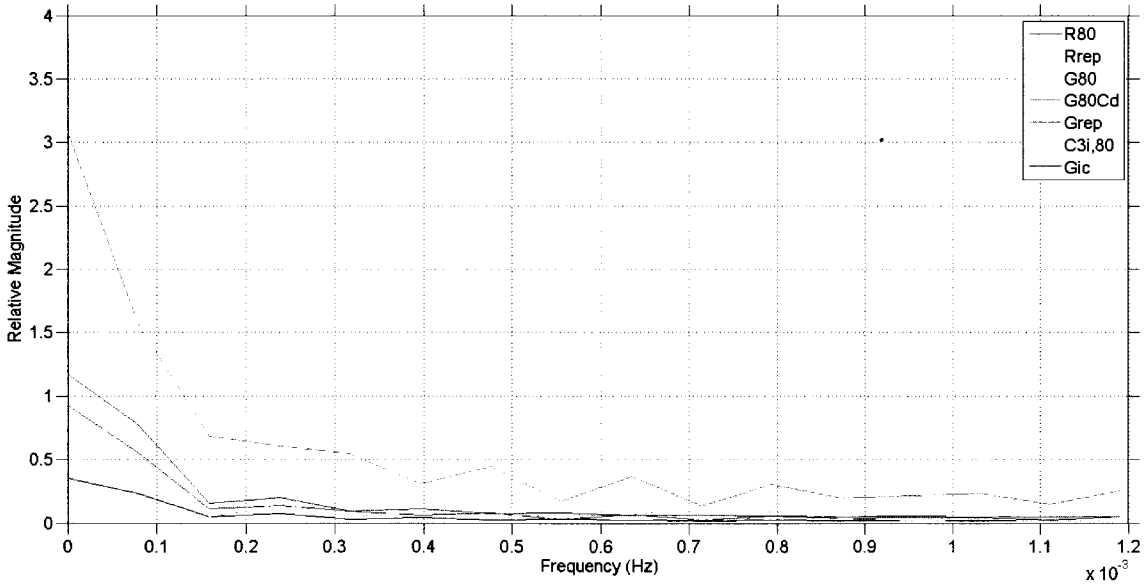


Figure 5.17: Spectrum of select GalSim species with TASS using the point spread function -30%,-20%,-10%,0,+10%,+20%,+30%. Values are normalized as per Section 3.4.1.

Since it is difficult to directly compare these figures to each other, two plots, each

showing all cases for a single species, have been produced. Figure 5.18 shows the $C_{3i;80}$ curves (which is subject to the greatest degree of aliasing and therefore shows the most visible effect) from each of the GalSim results above (as well as the no jitter case) plotted on the same set of axes. Similarly, Figure 5.19 shows G_{80} (which is not subject to as much aliasing, and expected to show little effect from TASS) in the same fashion. The results agree with these expectations: in the reference case at 7.1×10^{-4} Hz, the $C_{3i;80}$ magnitude is 3.916×10^{5}, but drops to 2.504×10^{5} when TASS is applied using the fifth point spread function. G_{80} shows much less variation, from 4.1×10^{-3} to 3.3×10^{-3}.

As with the jitter sampling results (Section 4.6), a conclusive determination that the intensity decrease results from alias suppression instead of generalized attenuation is not possible, but similar reasoning supports this view:

- TASS has been shown to suppress aliasing (Section 5.5.2);
- Aliasing is known to be occuring in this GalSim scenario (Section 3.5);
- The choice of TASS parameters affects decreases in signal intensity in a manner consistent with the effect of those parameters on alias suppression in the synthetic model (Figure 5.19, Section 5.5.5); and,
- A noticeable decrease in signal intensity is seen with $C_{3i;80}$ (which is aliased), whereas the minimally unaliased G_{80} is almost unaffected (Figure 5.18, Figure 5.19).

It is therefore reasonable to conclude that TASS is suppressing aliasing in GalSim.

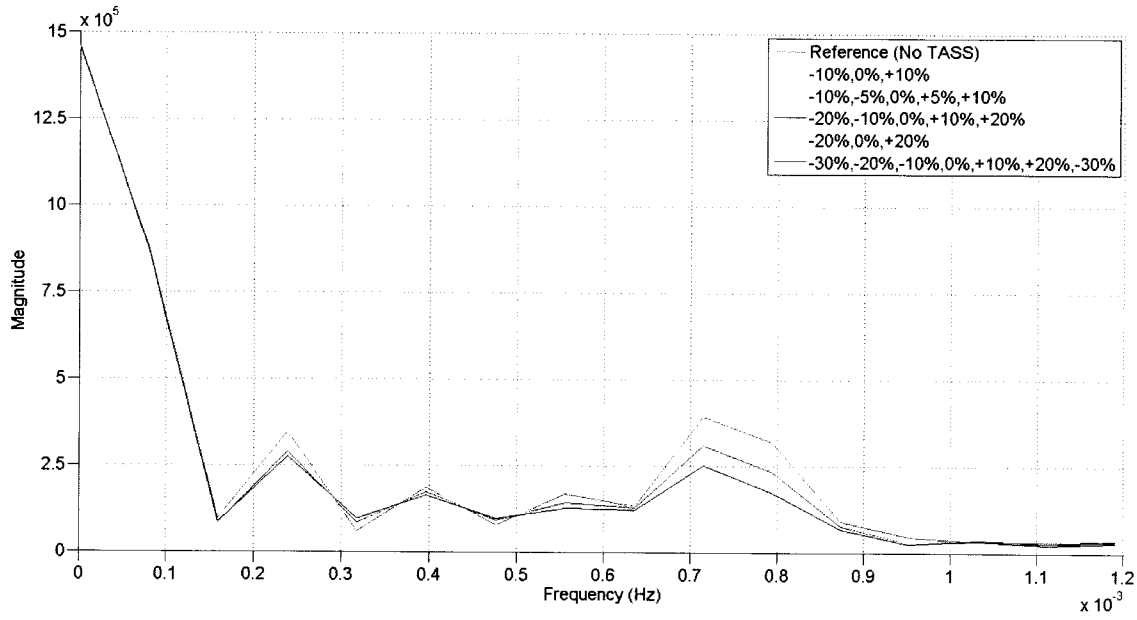


Figure 5.18: Comparison of $C_{3i;80}$ spectrum using various TASS schemes. "Reference" refers to the no-jitter case.

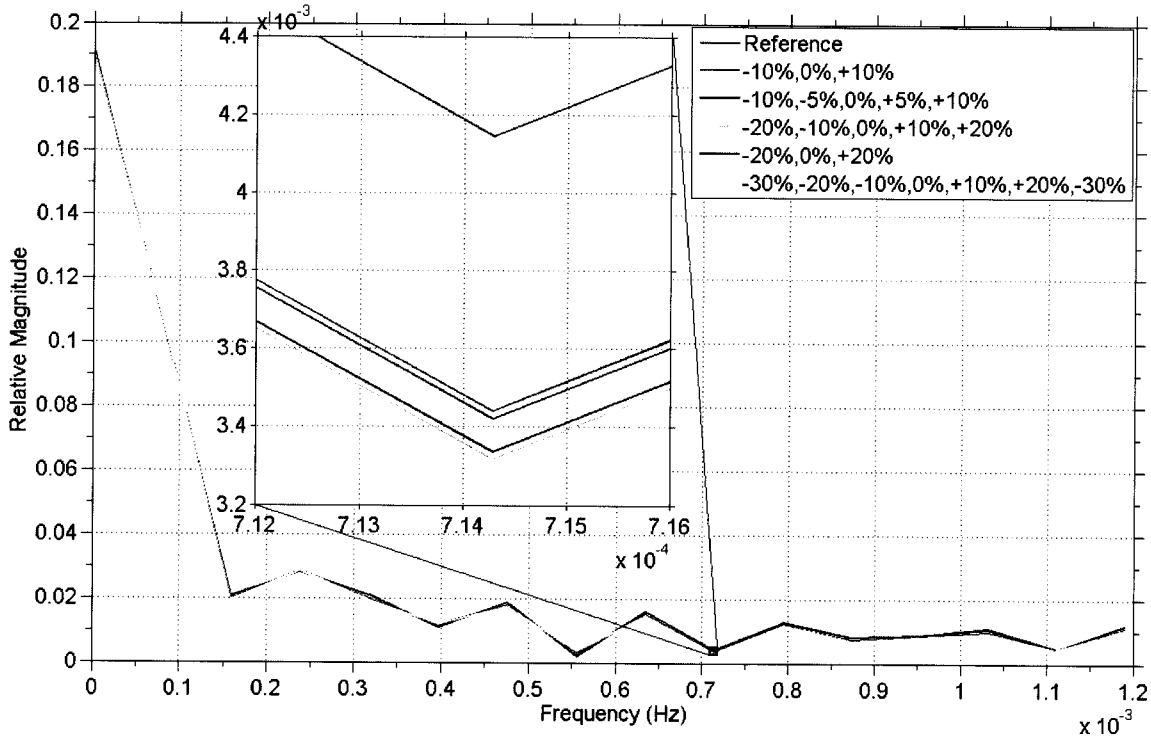


Figure 5.19: Comparison of G_{80} spectrum using various TASS schemes. "Reference" refers to the no-jitter case.

5.7 Conclusions

TASS is a promising technique which can be easily applied to TSGES with little or no increase in cost. Based on simulation results, it suppresses aliasing when used with biological systems at low sample numbers. It therefore has the potential to improve the accuracy of time-series gene expression data.

Chapter 6

Comparison of Jitter Sampling and Time Aggregation and Skip Sampling

6.1 Comparison of Degree of Alias Suppression

The most obvious basis on which jitter sampling and TASS may be compared is how well they fulfill the primary objective: suppression of aliasing. Unfortunately, the results are somewhat ambiguous. When comparing alias suppression in the synthetic data model, TASS appears to out-perform jitter sampling by a large margin: at 30 samples, the best TASS performance yielded a Q score of 0.01284, while jitter sampling was only able to achieve 0.2432. When evaluated against GalSim data, however, the ranking reverses: the highly aliased $C_{3i;80}$ signal drops to 1.403×10^5 under jitter sampling while TASS is only able to reduce this signal to 2.504×10^5.

These contradictory results may be due to different point spread functions and jitter distributions; it seems likely, though, that the performance of TASS and jitter sampling will depend at least in part on the nature of the system being measured. This implies that there is no single "ideal" choice between jitter sampling or TASS, or of the details of the chosen technique. Instead, the choice will vary depending on the particulars of the system under investigation.

6.2 Deleterious Effects

As discussed in Section 4.3, spectral leakage can be seen surrounding the primary peaks in the spectrum of the synthetic data model. (Note, particularly, Figure 4.4.)

When jitter sampling is applied, this spectral leakage is redistributed across the spectrum, essentially at random. This creates what is functionally equivalent to a noise floor, shown in Figure 4.5. A similar (but non-random) redistribution of the spectral leakage occurs with TASS; see Figure 5.4. The precise characteristics of this noise floor will vary depending on the system being measured and the choice of parameters selected for jitter sampling or TASS.

6.3 Predictability of Outcome

Jitter sampling is a stochastic process, and there is therefore an inherent element of unpredictability in its outcomes. The technical details were discussed in Chapter 4 (especially Section 4.4); here we consider the practical implications of this fact.

One of the major motivating factors for this work is the high cost of TSGESs; it would therefore be unacceptable to have an entire study "ruined" if this can be avoided. As a result of this consideration, techniques which are effective on a statistical basis but which can fail unpredictably may not be acceptable in real world contexts.

While jitter sampling has been shown to be an effective way of suppressing aliasing, its results are inherently random and any given run can produce undesireable results. From Figure ??, even 10% Gaussian jitter can result in $Q = 1.5$ in the worst observed result; jitter sampling has actually increased the intensity of the undesireable signal by 50% in this case. While the possibility exists to identify a "deterministic jitter" pattern (see Section 4.5) which would give predictable results, no such pattern has

yet been isolated. By contrast to jitter sampling, TASS is a deterministic process with predictable outcomes. Note that these concerns apply equally to the noise floor issue raised in the previous section.

Based on these factors, biologists conducting TSGES may well decline to implement jitter sampling and prefer TASS so as to avoid deleterious effects.

6.4 Reproducibility of Results

The concerns raised in Section 4.4 and Section 6.3 regarding the variability of the effect of jitter sampling also affect the reproducibility of the results. In general, two otherwise identical studies using different sets of jitter values drawn from the same statistical distribution cannot be expected to yield identical results. By contrast, all other things being equal, two different studies using TASS with the same point spread function should, in theory, have the same outcome. As reproducibility is a critical element of the scientific process, TASS may be preferred over jitter sampling; at a minimum, the report for a study using jitter sampling should include the actual sample timepoints, in addition to the nominal timepoints and jitter distribution.

6.5 Implementation Costs

As previously established, microarrays are normally the most expensive component of TSGES; jitter sampling and TASS should therefore have a negligible implementation cost in most situations. Under certain specific conditions, however, TASS may incur significant costs. If the collection of samples requires the euthanizing of an animal (e.g., collecting liver cells from a mouse), quintupling the required number of samples would require quintupling the number of animals used in the experiment; this may be prohibitive.

6.6 Specificity of Suppression

An important difference in the effects of jitter sampling and TASS relates to the frequency-domain specificity of the alias suppression. While both techniques essentially act as LPFs, supressing signal content above a given cut-off frequency, the determination of that cut-off frequency differs markedly between the two approaches.

Jitter sampling exclusively suppresses aliasing; the cut-off frequency is the Nyquist frequency and cannot be altered by the experimenter except by changing the sampling rate. By constrast, the cut-off frequency for TASS is governed by the point spread function. TASS could therefore fail to suppress some aliasing (if the effective cut-off is greater than the critical frequency) or suppress non-aliased desirable content (if the effective cut-off is too low).

6.7 Effect on Reverse Engineering Outcomes

The ideal basis for comparing the techniques would, of course, be their effect on the putative GRN reverse engineered from real or simulated TSGES data. Efforts were made to assess the presented techniques in this manner; these efforts were unsuccessful.

After some analysis, it was realized that only in the case where the GRN under study contained one or more measured species whose variation would be aliased to match the variation of a non-aliased measured species would the presence of aliasing affect the output of current reverse engineering algorithms, which work solely on the basis of correlation.

Given the extremely large number of species present in most real-world GRNs, it seems likely that this does occur "in the wild", but the absence of a suitable model for simulation containing such a case makes it impossible to prove in the current work.

6.8 Comprehensibility of Method

TASS is a relatively simple to understand process; it is (in terms of mental models) not far removed from time-moving averages, which are a basic signal processing technique. Jitter sampling, on the other hand, is not intuitive in implementation or effect. Biology researchers (who are generally not well-versed in signal processing concepts) may be reluctant to trust a critical study to a technique which they do not understand.

Chapter 7

Conclusions

7.1 Contributions

We have demonstrated three major points:

- Aliasing can occur in TSGES conducted using typical sampling rates from published studies;
- Jitter sampling can reduce aliasing in TSGES; and,
- TASS is a cost-effective technique capable of mitigating this aliasing.

Simulation of the GAL regulon (Chapter 3) showed that distortion of TSGES data from aliasing is possible in studies of biological systems, and that the most aggressive sampling interval found in the literature was insufficient to avoid this. The application of TASS and jitter sampling to the simulation showed that either of these techniques could mitigate this aliasing; additional testing of these techniques in a synthetic model confirmed that they are resilient to small sample numbers, as is common in TSGES.

7.1.1 Novelty

As discussed in Section 2.3, a number of different strategies for making the best use of the limited number of measurements available in TSGESs have been developped. While a number of these strategies considered the issue of noise, none appeared to expressly deal with the issue of aliasing. The only mention of aliasing in the context of TSGES was found in [39], which cited [40] and [41] as alternative techniques, but dismissed them as inapplicable to a biological context. According to [39]

> the authors *[of [41]]* aim to estimate a wide spectral range of frequencies of a non uniformly sampled signal. Their approach is, however, aimed more at real-time applications and longer signals than those usually present in microarray studies.

These objections apply equally to all other implementations and discussions of jitter sampling discovered in the course of this work. The contributions discussed herein are therefore, to the best of the author's knowledge, unique in applying jitter sampling under constraints to the short sample sequences typical of TSGESs.

While time-moving averages are a well-known and pervasive signal processing technique, the more complex TASS is almost unknown in the literature. "Time aggregation" and "skip sampling" do appear in some econometrics papers, but as separate alternatives, not combined as a single process. In [35], "time aggregation" is defined as

> summing high frequency time serie.s data into low frequency data, as for instance, the aggregation of monthly data into quarterly or annual data.

and "skip sampling"

> occurs when data are only observable every certain time period.

In that article, and most others in the field, such as [36] and [40] , both TA and SS are regarded as deleterious but necessary processes. Research on this area focuses on statistical properties of and relationships between economic and demographic time series data, normally at time scales in the range of months to years.

None of the articles discovered in the literature review showed any deliberate application of TASS - in all cases, the presence of time aggregation or skip sampling was viewed as unavoidable limitation of the source data. Thus, the present work is apparently unique in deliberately applying TASS to improve the usability of time series data.

7.2 Future Work

The contributions detailed in this work suggest several avenues for future research. These opportunities can be broken down into three main categories: looking for existing effects; analyzing and optimizing the techniques; and, validating the techniques.

The techniques described in this work (TASS and jitter sampling) were originally motivated by the suspicion that they might be inherent in the way TSGES were conducted (see Section 4.1.2 and Section 5.3). It would be informative to investigate the lab procedure for TSGES in further detail and determine if this initial suspicion is justified.

While this work did investigate the feasibility of applying TASS and jitter sampling to TSGES, it did not seek to determine optimal usage. The application of jitter sampling requires consideration of the alias suppression effect in light of the potential to distort the measured signal; further testing is required to determine the best way to manage this tradeoff. Additionally, it may be possible to identify a specific sequence of pre-determined, non-stochastic jitter values which would yield optimum results. The factors governing the relationship between the choice of point spread function and

the results of TASS are not yet well understood, and requires further investigation. Additionally, no direct comparison of jitter sampling and TASS was done; a head-to-head trial could be highly informative.

Finally, all of the work done to date has been done *in silico*, that is, in computer simulations. In order to truly validate these techniques for use with TSGES, it would be necessary to conduct *in vivo* or *in vitro* studies with live organisms in a properly equipped wet-lab.

7.3 Recommendations

Given that aliasing has been shown to be a plausible impediment to TSGES, biology researchers should consider taking steps to mitigate its effects. Both techniques described in this work are viable options; the choice between them, and the choice of parameters for the selected technique will depend on the details of the experiment and the researcher's objectives. Ideally, a number of different permutations would be tested against simulated data from the biological system under study in order to determine the most appropriate choice; in practice, suitable simulations are unlikely to be available, and the best obtainable substitute (multiple superimposed sinusoids, band-limited pink noise or band-limited pink noise are obvious candidates) should be used. Consideration should also be given to the relative importance of accuracy of the suppression cut-off, reproducibility of the experiment, comprehensibility of jitter sampling to the audience and any study-specific factors.

List of References

[1] J. S. Choinski, "RNA and protein synthesis," December 2008. http://faculty.uca.edu/johnc/rnaprot1440.htm.

[2] Q. Tian, S. B. Stepaniants, M. Mao, L. Weng, M. C. Feetham, M. J. Doyle, E. C. Yi, H. Dai, V. Thorsson, J. Eng, D. Goodlett, J. P. Berger, B. Gunter, P. S. Linseley, R. B. Stoughton, R. Aebersold, S. J. Collins, W. A. Hanlon, and L. E. Hood, "Integrated genomic and proteomic analyses of gene expression in mammalian cells," *Mol Cell Proteomics*, vol. 3, no. 10, pp. 960–969, 2004.

[3] S. P. Gygi, Y. Rochon, B. R. Franza, and R. Aebersold, "Correlation between protein and mRNA abundance in yeast," *Mol. Cell. Biol.*, vol. 19, no. 3, pp. 1720–1730, 1999.

[4] L. Nie, G. Wu, and W. Zhang, "Correlation between mRNA and protein abundance in desulfovibrio vulgaris: A multiple regression to identify sources of variations," *Biochemical and Biophysical Research Communications*, vol. 339, no. 2, pp. 603–610, 2006.

[5] R. Brockmann, A. Beyer, J. J. Heinisch, and T. Wilhelm, "Posttranscriptional expression regulation: What determines translation rates?," *PLoS Comput Biol*, vol. 3, p. e57, Mar 2007.

[6] N. Geva-Zatorsky, N. Rosenfeld, S. Itzkovitz, R. Milo, A. Sigal, E. Dekel, T. Yarnitzky, Y. Liron, P. Polak, G. Lahav, and U. Alon, "Oscillations and variability in the p53 system," *Mol. Syst. Biol.*, vol. 2, Jun 2006.

[7] W. Zhao, K. Agyepong, E. Serpedin, and E. Dougherty, "Identifying drosophila cell-cycle regulated genes from irregular microarray data," *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pp. 633–636, April 2008.

[8] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher, "Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization," *Mol. Biol. Cell*, vol. 9, no. 12, pp. 3273–3297, 1998.

[9] R. J. Cho, M. J. Campbell, E. A. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. G. Wolfsberg, A. E. Gabrielian, D. Landsman, D. J. Lockhart, and R. W. Davis, "A genome-wide transcriptional analysis of the mitotic cell cycle," *Mol. Cell.*, vol. 2, pp. 65–73, July 1998.

[10] G. Zhu, P. T. Spellman, T. Volpe, P. O. Brown, D. Botstein, T. N. Davis, and B. Futcher, "Two yeast forkhead genes regulate the cell cycle and pseudohyphal growth," *Nature*, vol. 406, pp. 90–94, July 2000.

[11] A. V. Werhli, M. Grzegorczyk, and D. Husmeier, "Comparative evaluation of reverse engineering gene regulatory networks with relevance networks, graphical gaussian models and Bayesian networks," *Bioinformatics*, vol. 22, no. 20, pp. 2523–2531, 2006.

[12] C.-C. Wu, H.-C. Huang, H.-F. Juan, and S.-T. Chen, "GeneNetwork: An interactive tool for reconstruction of genetic networks using microarray data," *Bioinformatics*, vol. 20, no. 18, pp. 3691–3693, 2004.

[13] A. J. Butte, L. Bao, B. Y. Reis, T. W. Watkins, and I. S. Kohane, "Comparing the similarity of time-series gene expression using signal processing metrics," *Comput. Biomed. Res.*, vol. 34, no. 6, pp. 396–405, 2001.

[14] C. Mller-Levet, F. Klawonn, K.-H. Cho, H. Yin, and O. Wolkenhauer, "Clustering of unevenly sampled gene expression time-series data," *Fuzzy Sets and Systems*, vol. 152, pp. 49–66, 5 2005.

[15] Z. Bar-Joseph, D. K. Gifford, and T. S. Jaakkola, "Fast optimal leaf ordering for hierarchical clustering," *Bioinformatics*, vol. 17, no. Supp 1, pp. S22–29, 2001.

[16] Z. Bar-Joseph, E. D. Demaine, D. K. Gifford, N. Srebro, A. M. Hamel, and T. S. Jaakkola, "K-ary clustering with optimal leaf ordering for gene expression data," *Bioinformatics*, vol. 19, no. 9, pp. 1070–1078, 2003.

[17] K. Y. Yeung, D. R. Haynor, and W. L. Ruzzo, "Validating clustering for gene expression data," *Bioinformatics*, vol. 17, no. 4, pp. 309–318, 2001.

[18] Wikipedia, "Lac operon — wikipedia, the free encyclopedia," 2008. [Online; accessed 18-November-2008].

[19] Wikipedia, "Thiazolidinedione — wikipedia, the free encyclopedia," 2009. [Online; accessed 21-May-2009].

[20] A. Hsiao, D. S. Worrall, J. M. Olefsky, and S. Subramaniam, "Variance-modeled posterior inference of microarray data: Detecting gene-expression changes in 3T3-L1 adipocytes," *Bioinformatics*, vol. 20, no. 17, pp. 3108–3127, 2004.

[21] W. C. Winkelmayer, S. Setoguchi, R. Levin, and D. H. Solomon, "Comparison of cardiovascular outcomes in elderly patients with diabetes who initiated rosiglitazone vs pioglitazone therapy," *Arch Intern Med*, vol. 168, no. 21, pp. 2368–2375, 2008.

[22] D. Savransky, "Lomb (Lomb-Scargle) periodogram," May 2008. [Available on MatlabCentral as lomb.m.].

[23] C. Shannon, "Communication in the presence of noise," *Proceedings of the IRE*, vol. 37, pp. 10–21, Jan 1949. As reprinted in [42].

[24] Z. Bar-Joseph, G. K. Gerber, D. K. Gifford, T. S. Jaakkola, and I. Simon, "Continuous representations of time-series gene expression data," *Journal of Computational Biology*, vol. 10, no. 3-4, pp. 341–356, 2003.

[25] I. Simon, Z. Siegfried, J. Ernst, and Z. Bar-Joseph, "Combined static and dynamic analysis for determining the quality of time-series expression profiles," *Nat Biotech*, vol. 23, pp. 1503–1508, Dec 2005.

[26] R. Singh, N. Palmer, D. Gifford, B. Berger, and Z. Bar-Joseph, "Active learning for sampling in time-series experiments with application to gene expression analysis," in *ICML '05: Proceedings of the 22nd international conference on Machine learning*, (New York, NY, USA), pp. 832–839, ACM, 2005.

[27] R. Singh, N. Palmer, D. Gifford, B. Berger, and Z. Bar-Joseph, "An active learning approach for appropriate sampling during time-series expression experiments," in *Conference on Research in Computational Molecular Biology*, 2005.

[28] T. Ideker, V. Thorsson, J. A. Ranish, R. Christmas, J. Buhler, J. K. Eng, R. Bumgarner, D. R. Goodlett, R. Aebersold, and L. Hood, "Integrated genomic and proteomic analyses of a systematically perturbed metabolic network," *Science*, vol. 292, no. 5518, pp. 929–934, 2001.

[29] S. A. Ramsey, J. J. Smith, D. Orrell, M. Marelli, T. W. Petersen, P. de Atauri, H. Bolouri, and J. D. Aitchison, "Dual feedback loops in the gal regulon suppress

cellular heterogeneity in yeast," *Nat Genet*, vol. 38, pp. 1082–1087, September 2006.

[30] V. Abedi. Personal communication, January 2008.

[31] T. Pramila, S. Miles, D. GuhaThakurta, D. Jemiolo, and L. L. Breeden, "Conserved homeodomain proteins interact with MADS box protein Mcm1 to restrict ECB-dependent transcription to the M/G1 phase of the cell cycle," *Genes & Development*, vol. 16, no. 23, pp. 3034–3045, 2002.

[32] Z. Bar-Joseph, "Analyzing time series gene expression data," *Bioinformatics*, vol. 20, no. 16, pp. 2493–2503, 2004.

[33] I. Bilinskis and A. K. Mikelson, *Randomized Signal Processing.* Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1992.

[34] A. Tarczynski and N. Allay, "Spectral analysis of randomly sampled signals: suppression of aliasing and sampler jitter," *Signal Processing, IEEE Transactions on*, vol. 52, no. 12, pp. 3324–3334, Dec. 2004.

[35] W. Hu, "Time aggregation and skip sampling in cointegration tests," *Statistical Papers*, vol. 37, pp. 225–234, September 1996.

[36] L. P. Hansen and T. J. Sargent, "An appreciation of A.W. Phillips," 1995.

[37] P. Banerjee. Personal communication, October 2007.

[38] Z. Bar-Joseph, S. Farkash, D. K. Gifford, I. Simon, and R. Rosenfeld, "Deconvolving cell cycle expression data with complementary information," *Bioinformatics*, vol. 20, no. suppl_1, pp. i23–30, 2004.

[39] M. Ahdesmaki, H. Lahdesmaki, A. Gracey, l. Shmulevich, and O. Yli-Harja, "Robust regression for periodicity detection in non-uniformly sampled time-course gene expression data," *BMC Bioinformatics*, vol. 8, no. 1, p. 233, 2007.

[40] A. Tarczynski and D. Qu, "Optimal periodic sampling sequences for nearly-alias-free digital signal processing," in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pp. 1425 – 1428 Vol. 2, 23-26 2005.

[41] A. Tarczynski, D. Bland, and T. Laakso, "Spectrum estimation of non-uniformly sampled signals," *Industrial Electronics, 1996. ISIE '96., Proceedings of the IEEE International Symposium on*, vol. 1, pp. 196–200 vol.1, 17-20 Jun 1996.

[42] C. Shannon, "Communication in the presence of noise," *Proceedings of the IEEE*, vol. 86, pp. 447–457, Feb 1998. [Reprint].

Appendix A

GAL Regulon Model

The following set of 19 ordinary differential equations govern and define the behaviour of the simulated GAL regulon introduced in Chapter 3 used throughout this work. This model was adapted from [29] by [30].

$$
\begin{aligned}
\frac{dR_1}{dt} &= K_{ir,gal1}\frac{A}{B} - K_{dr,gal1}R_1 \\
A &= 4K_pG_4 + 12K_qG_{80d}(K_pG_4)^2 + 6(K_pG_4)^2 + 4(K_pG_4)^3 + 12K_qG_{80d}(K_pG_4)^3 \\
&\quad + 12q_r(K_pG_4)^3(K_qG_{80d})^2 + (K_pG_4)^4 + 4K_qG_{80d}(K_pG_4)^4 \\
&\quad + 4q_r^2(K_pG_4)^4(K_qG_{80d})^3 + 6q_r(K_pG_4)^4(K_qG_{80d})^2 \\
B &= \Big(4K_pG_4 + 12K_qG_{80d}(K_pG_4)^2 + 6(K_p * G_4)^2 + 4(K_pG_4)^3 + 12K_qG_{80d}(K_pG_4)^3 \\
&\quad + 12q_r(K_pG_4)^3(K_qG_{80d})^2 + (K_pG_4)^4 + 4K_qG_{80d}(K_pG_4)^4 \\
&\quad + 4{q_r}^2(K_pG_4)^4(K_qG_{80d})^3 + 6q_r(K_pG_4)^4(K_qG_{80d})^2\Big) \\
&\quad + 1 + 4K_pG_4K_qG_{80d} + 6q_r(K_pG_4)^2(K_qG_{80d})^2 \\
&\quad + 4{q_r}^2(K_pG_4)^3(K_qG_{80d})^3 + {q_r}^3(K_pG_4)^4(K_qG_{80d})^4
\end{aligned}
\tag{A.1}
$$

Element	Value
R1	0.2647
R2	0.3305
R3	0.9044
R4	0.2647
R80	1.1871
Rrep	132.3267
G1	1156.7
G2	4341.2
G3	0
G3i	0.1563
G4	308.92
G4d	132.327
G80	0.1138
G80C	0.1095
G80d	157.229
G80Cd	157.229
Grep	0
C3i;80	0
Gic	0

Table A.1: Initial Model Values

$$\frac{dR_2}{dt} = K_{ir,gal2}\frac{A}{B} - K_{dr,gal2}R_2$$

$$\begin{aligned}
A =& 5K_pG_4 + 10(K_pG_4)^2 + 20K_qG_{80d}(K_pG_4)^2 + 30q_r(K_pG_4)^3(K_qG_{80d})^2 \\
&+ 30(K_p * G_4)^3(K_qG_{80d}) + 10(K_pG_4)^3 + 5(K_pG_4)^4 + 30q_r(K_pG_4)^4(K_qG_{80d})^2 \\
&+ 20K_qG_{80d}(K_pG_4)^4 + 20q_r^2(K_pG_4)^4(K_qG_{80d})^3 + (K_pG_4)^5 + 5K_q * G_{80d}(K_pG_4)^5 \\
&+ 5q_r^3(K_p * G_4)^5(K_qG_{80d})^4 + 10q_r^2(K_pG_4)^5(K_qG_{80d})^3 + 10q_r(K_pG_4)^5(K_qG_{80d})^2 \\
B =& \Big(5(K_pG_4) + 10(K_pG_4)^2 + 20(K_pG_4)^2(K_qG_{80d}) + 30q_r(K_pG_4)^3(K_qG_{80d})^2 \\
&+ 30(K_pG_4)^3(K_qG_{80d}) + 10(K_pG_4)^3 + 5(K_pG_4)^4 + 30q_r(K_pG_4)^4(K_qG_{80d})^2 \\
&+ 20K_qG_{80d}(K_pG_4)^4 + 20q_r^2(K_p * G_4)^4(K_q * G_{80d})^3 + (K_pG_4)^5 \\
&+ 5K_qG_{80d}(K_pG_4)^5 + 5q_r^3(K_pG_4)^5(K_qG_{80d})^4 + 10q_r^2(K_pG_4)^5(K_qG_{80d})^3 \\
&+ 10q_r(K_pG_4)^5(K_qG_{80d})^2\Big) + 1 + 5K_pG_4K_qG_{80d} + 10q_r(K_pG_4)^2(K_qG_{80d})^2 \\
&+ 10q_r^2(K_pG_4)^3(K_qG_{80d})^3 + 5q_r^3(K_pG_4)^4(K_qG_{80d})^4 + q_r^4(K_p * G_4)^5(K_qG_{80d})^5
\end{aligned} \tag{A.2}$$

$$\frac{dR_3}{dt} = K_{ir,gal3}q_3\frac{K_pG_4}{1 + K_pG_4 + K_pG_4K_qG_{80d}} - K_{dr,gal3}R_3 \tag{A.3}$$

$$
\begin{aligned}
\frac{dR_4}{dt} &= K_{ir,rep}\frac{A}{B} - K_{dr,rep}R_4 \\
A &= 4K_pG_4 + 12K_qG_{80d}(K_pG_4)^2 + 6(K_pG_4)^2 \\
&\quad + 4(K_pG_4)^3 + 12(K_pG_4)^3K_qG_{80d} + 12q_r(K_pG_4)^3(K_qG_{80d})^2 + (K_pG_4)^4 \\
&\quad + 4K_qG_{80d}(K_pG_4)^4 + 4q_r^2(K_pG_4)^4(K_qG_{80d})^3 + 6q_r(K_pG_4)^4(K_qG_{80d})^2 \\
B &= (4(K_pG_4) + 12(K_pG_4)^2(K_qG_{80d}) + 6(K_pG_4)^2 + 4(K_pG_4)^3 \\
&\quad + 12(K_pG_4)^3(K_qG_{80d}) + 12q_r(K_pG_4)^3(K_qG_{80d})^2 + (K_pG_4)^4 \\
&\quad + 4(K_pG_4)^4(K_qG_{80d}) + 4q_r^2(K_pG_4)^4(K_qG_{80d})^3 \\
&\quad + 6q_r(K_pG_4)^4(K_qG_{80d})^2) + 1 + 4(K_pG_4)(K_qG_{80d}) \\
&\quad + 6q_r(K_pG_4)^2(K_qG_{80d})^2 + 4q_r^2(K_pG_4)^3(K_qG_{8q_r})^3 \\
&\quad + q_r^3(K_pG_4)^4(K_qG_{80d})^4
\end{aligned}
\tag{A.4}
$$

$$
\frac{dR_{80}}{dt} = K_{ir,gal80}\frac{K_pG_4}{1 + K_pG_4 + K_pG_4K_qG_{80d}} - K_{dr,gal80}R_{80} \tag{A.5}
$$

$$
\frac{dR_{rep}}{dt} = K_{ip,gal1}R_1 - K_{dp,gal1}R_{rep} \tag{A.6}
$$

$$
\frac{dG_1}{dt} = K_{ip,gal2}R_2 - K_{dp,gal2}G_1 \tag{A.7}
$$

$$\frac{dG_2}{dt} = K_{ip,gal3}R_3 - K_{dp,gal3}G_3 - K_{fi}G_3G_{ic} + K_{ri}G_3 \tag{A.8}$$

$$\frac{dG_3}{dt} = K_{fi}G_3G_{ic} - K_{ri}G_3 - K_{dp,gal3}G_3 - K_{fd,3i,80}G_{80Cd}G_3 + K_{dr,3i,80}G_{rep} \tag{A.9}$$

$$\frac{dG_{3i}}{dt} = K_{ip,gal4}G_{ic} - K_{dp,gal4}G_{3i} - 2K_{fd}G_{3i}G_{3i} + 2K_{rd}G_4 \tag{A.10}$$

$$\frac{dG_4}{dt} = K_{fd}G_{3i}^2 - K_{rd}G_4 - K_{dp,gal4}G_4 \tag{A.11}$$

$$\frac{dG_{4d}}{dt} = K_{ip,rep}R_4 - K_{dp,rep}G_{4d} \tag{A.12}$$

$$\frac{dG_{80}}{dt} = -K_{dp,gal80}G_{80} - K_{f80}G_{80} + K_{r80}G_{80C} - 2K_{fd}G_{80}G_{80} + 2K_{rd}G_{80d} \tag{A.13}$$

$$\begin{aligned}\frac{dG_{80C}}{dt} = {} & K_{ip,gal80}R_{80} + K_{f80}G_{80} - K_{r80}G_{80C} - 2K_{fd}G_{80C}G_{80C} \\ & + 2K_{rd}G_{80Cd} - K_{dp,gal80}G_{80C}\end{aligned} \tag{A.14}$$

$$\frac{dG_{80d}}{dt} = K_{fd}G_{80}^2 - K_{rd}G_{80d} - K_{dp,gal80}G_{80d} - K_{f80}G_{80d} + K_{r80}G_{80Cd} \tag{A.15}$$

$$\begin{aligned}\frac{dG_{80Cd}}{dt} = {} & K_{fd}G_{80C}^2 - K_{rd}G_{80Cd} - K_{dp,gal80}G_{80Cd} + K_{f80}G_{80d} \\ & - K_{r80}G_{80Cd} - K_{fd,3i,80}G_{80Cd}G_3K_{dr,3i,80}G_{rep}\end{aligned} \tag{A.16}$$

$$\frac{dG_{rep}}{dt} = K_{fd,3i,80}G_{80Cd}G_3 - K_{dr,3i,80}G_{rep} - \frac{1}{2}K_{dp,gal3}G_{rep} \tag{A.17}$$

$$\begin{aligned}\frac{dG_{3i,80}}{dt} = {} & \left(\frac{K_{tr}G_2(G_{ex} - 4.65 \times 10^{-}8G_{ic})}{K_{mtr} + G_{ex} + (4.65 \times 10^{-8})G_{ic} + (4.65 \times 10^{-8})\frac{A_{tr}}{K_{mtr}}G_{ex}G_{ic}}\right) \\ & - \left(\frac{K_{cat,GK}G_1G_{ic}}{K_{m,GK} + G_{ic}}\right) - K_{fi}G_3G_{ic} + K_{ri}G_3\end{aligned} \tag{A.18}$$

$$\frac{dG_{ic}}{dt} = K_{ir,gal4} - G_{ic}K_{dr,gal4} \tag{A.19}$$

Appendix B

GalSim Source Code Listing

Listing B.1: A typical invocation of GalSim

```
>> [t y]=run_sim ([0:7:119],square_food(0.1,5))
```

Listing B.2: run_sim.m

```
function [t, y] = run_sim(tspan,feed)


global par;
global feeding;


feeding=@(t)feed(t);
u=get_init_values();
par=get_params();


[t,y] = ode45('rate_eqs',tspan,u);
t=transpose(t);
y=transpose(y);
```

Listing B.3: randomize_sampling.m

```
function r_tspan = randomize_sampling(tspan, amt, r_fct)


r_tspan=tspan;
r_fct('state',sum(100*clock));


for i=2:size(tspan,2)
        t_i=tspan(i);
        t_prev=tspan(i-1);
        del_ti=t_i-t_prev;
        shift=del_ti*amt*r_fct();
        r_tspan(i)=r_tspan(i)+shift;
end
```

Listing B.4: sine_food.m

```
function f = square_food(A,T)
f=@(x)A*((floor(x/T)/2)==floor(floor(x/T)/2));
```

Listing B.5: get_init_values.m

```
function u = get_init_values()
%INIT_VALUES Summary of this function goes here
%   Detailed explanation goes here


%DEFAULT INITIAL CONDITION
u = [    0.2647            0.3305            0.9044
   0.2647            1.1871   ...
        132.3267          1156.7            4341.2              0
                            0.1563   ...
```

```
        308.92          132.327          0.1138
          0.1095            157.229  ...
        157.229          0                0                0
                              ];
```

Listing B.6: get_params.m

```
function param = get_params()


%DEFAULT PARAMETERS
    param(1) = 0.7379;
    param(2) = 2.542 ;
    param(3) = 0.7465 ;
    param(4) = 0.009902;
    param(5) = 0.6065 ;
    param(6) = 1.1440;
    param(7) = 0.571 ;
    param(8) = 0.02104;
    param(9) = 0.1052;
    param(10) = 30 ;
    param(11) = 1.9254 ;
    param(12) = 13.4779 ;
    param(13) = 55.4518 ;
    param(14) = 10.7091 ;
    param(15) = 3.6737 ;
    param(16) = 5.7762 ;
    param(17) = 0.02236 ;
    param(18) = 0.07702 ;
```

```
param(19) = 0.02666 ;
param(20) = 0.02476 ;
param(21) = 0.02888;
param(22) = 0.03466  ;
param(23) = 0.003851;
param(24) = 0.003851;
param(25) = 0.01155 ;
param(26) = 0.006931 ;
param(27) = 0.006931 ;
param(28) = 0.01155 ;
param(29) = 100 ;
param(30) = 0.001;
param(31) = 0.000000745 ;
param(32) = 890;
param(33) = 50;
param(34) = 50 ;
param(35) = 0.02572;
param(36) = 0.01596;
param(37)= 1 ;
param(38)=1.0 ;
param(39)=4350;
param(40)=3350;
param(41)=12903000;
param(42)=0.0001;
```

Listing B.7: rate_eqs.m

```
function f = rate_eqs(t,u);
```

```
global par
global feeding

%%Parameters
        Kirgal1 = par(1);
        Kirgal2 = par(2);
        Kirgal3 = par(3);
        Kirgal4 = par(4);
        Kirgal80 = par(5);
        Kirrep = par(6);
        q3 = par(7);
        Kp = par(8);
        Kq = par(9);
        qr = par(10);
        Kipgal1 = par(11);
        Kipgal2 = par(12);
        Kipgal3 = par(13);
        Kipgal4 = par(14);
        Kipgal80 = par(15);
        Kiprep = par(16);
        Kdrgal1 = par(17);
        Kdrgal2 = par(18);
        Kdrgal3 = par(19);
        Kdrgal4 = par(20);
        Kdrgal80 = par(21);
        Kdrrep = par(22);
        Kdpgal1 = par(23);
```

```
        Kdpgal2 = par(24);
        Kdpgal3 = par(25);
        Kdpgal4 = par(26);
        Kdpgal80 = par(27);
        Kdprep = par(28);
        Kfd = par(29);
        Krd = par(30);
        Kfi = par(31);
        Kri = par(32);
        Kf80 = par(33);
        Kr80 = par(34);
        Kfd3i80 = par(35);
        Kdr3i80 = par(36);
        Atr = par(37);
        Kmtr = par(38);
        Ktr = par(39);
        Kcatgk = par(40);
        Kmgk = par(41);
%         Gex = par(42);


%%Feeding Time
Gex=feeding(t);

%%Equations
     f = zeros(19,1);      % a column vector
```

```
f(1) = Kirgal1*((4*(Kp*u(11) )+12*(Kp*u(11) )^2*(Kq*u(15)
    )+6*(Kp*u(11) )^2+4*(Kp*u(11) )^3+12*(Kp*u(11) )^3*(
   Kq*u(15) )+12*qr*(Kp*u(11) )^3*(Kq*u(15) )^2+(Kp*u(11)
    )^4+4*(Kp*u(11) )^4*(Kq*u(15) )+4*qr^2*(Kp*u(11) )
   ^4*(Kq*u(15) )^3+6*qr*(Kp*u(11) )^4*(Kq*u(15) )^2)
   /((4*(Kp*u(11) )+12*(Kp*u(11) )^2*(Kq*u(15) )+6*(Kp*u
   (11) )^2+4*(Kp*u(11) )^3+12*(Kp*u(11) )^3*(Kq*u(15) )
   +12*qr*(Kp*u(11) )^3*(Kq*u(15) )^2+(Kp*u(11) )^4+4*(Kp
   *u(11) )^4*(Kq*u(15) )+4*qr^2*(Kp*u(11) )^4*(Kq*u(15)
   )^3+6*qr*(Kp*u(11) )^4*(Kq*u(15) )^2)+1+4*(Kp*u(11) )
   *(Kq*u(15) )+6*qr*(Kp*u(11) )^2*(Kq*u(15) )^2+4*qr^2*(
   Kp*u(11) )^3*(Kq*u(15) )^3+qr^3*(Kp*u(11) )^4*(Kq*u
   (15) )^4))-Kdrgal1*u(1) ;
```

```
f(2) = Kirgal2*((5*(Kp*u(11) )+10*(Kp*u(11) )^2+20*(Kp*u
    (11) )^2*(Kq*u(15) )+30*qr*(Kp*u(11) )^3*(Kq*u(15) )
    ^2+30*(Kp*u(11) )^3*(Kq*u(15) )+10*(Kp*u(11) )^3+5*(Kp
    *u(11) )^4+30*qr*(Kp*u(11) )^4*(Kq*u(15) )^2+20*(Kp*u
    (11) )^4*(Kq*u(15) )+20*qr^2*(Kp*u(11) )^4*(Kq*u(15) )
    ^3+(Kp*u(11) )^5+5*(Kp*u(11) )^5*(Kq*u(15) )+5*qr^3*(
    Kp*u(11) )^5*(Kq*u(15) )^4+10*qr^2*(Kp*u(11) )^5*(Kq*u
    (15) )^3+10*qr*(Kp*u(11) )^5*(Kq*u(15) )^2)/((5*(Kp*u
    (11) )+10*(Kp*u(11) )^2+20*(Kp*u(11) )^2*(Kq*u(15) )
    +30*qr*(Kp*u(11) )^3*(Kq*u(15) )^2+30*(Kp*u(11) )^3*(
    Kq*u(15) )+10*(Kp*u(11) )^3+5*(Kp*u(11) )^4+30*qr*(Kp*
    u(11) )^4*(Kq*u(15) )^2+20*(Kp*u(11) )^4*(Kq*u(15) )
    +20*qr^2*(Kp*u(11) )^4*(Kq*u(15) )^3+(Kp*u(11) )^5+5*(
    Kp*u(11) )^5*(Kq*u(15) )+5*qr^3*(Kp*u(11) )^5*(Kq*u
    (15) )^4+10*qr^2*(Kp*u(11) )^5*(Kq*u(15) )^3+10*qr*(Kp
    *u(11) )^5*(Kq*u(15) )^2)+1+5*(Kp*u(11) )*(Kq*u(15) )
    +10*qr*(Kp*u(11) )^2*(Kq*u(15) )^2+10*qr^2*(Kp*u(11) )
    ^3*(Kq*u(15) )^3+5*qr^3*(Kp*u(11) )^4*(Kq*u(15) )^4+qr
    ^4*(Kp*u(11) )^5*(Kq*u(15) )^5))-Kdrgal2*u(2) ;
f(3) = Kirgal3*q3*((Kp*u(11) )/(1+(Kp*u(11) )+(Kp*u(11) )
    *(Kq*u(15) )))-Kdrgal3*u(3) ;
```

```
f(4) = Kirrep*((4*(Kp*u(11) )+12*(Kp*u(11) )^2*(Kq*u(15)
    )+6*(Kp*u(11) )^2+4*(Kp*u(11) )^3+12*(Kp*u(11) )^3*(Kq
    *u(15) )+12*qr*(Kp*u(11) )^3*(Kq*u(15) )^2+(Kp*u(11) )
    ^4+4*(Kp*u(11) )^4*(Kq*u(15) )+4*qr^2*(Kp*u(11) )^4*(
    Kq*u(15) )^3+6*qr*(Kp*u(11) )^4*(Kq*u(15) )^2)/((4*(Kp
    *u(11) )+12*(Kp*u(11) )^2*(Kq*u(15) )+6*(Kp*u(11) )
    ^2+4*(Kp*u(11) )^3+12*(Kp*u(11) )^3*(Kq*u(15) )+12*qr
    *(Kp*u(11) )^3*(Kq*u(15) )^2+(Kp*u(11) )^4+4*(Kp*u(11)
     )^4*(Kq*u(15) )+4*qr^2*(Kp*u(11) )^4*(Kq*u(15) )^3+6*
    qr*(Kp*u(11) )^4*(Kq*u(15) )^2)+1+4*(Kp*u(11) )*(Kq*u
    (15) )+6*qr*(Kp*u(11) )^2*(Kq*u(15) )^2+4*qr^2*(Kp*u
    (11) )^3*(Kq*u(15) )^3+qr^3*(Kp*u(11) )^4*(Kq*u(15) )
    ^4))-Kdrrep*u(4);
f(5) = Kirgal80*((Kp*u(11) )/(1+(Kp*u(11) )+(Kp*u(11) )*(
    Kq*u(15) )))-Kdrgal80*u(5) ;
f(6) = Kipgal1*u(1) -Kdpgal1*u(6) ;
f(7) = Kipgal2*u(2) -Kdpgal2*u(7) ;
f(8) = Kipgal3*u(3) -Kdpgal3*u(8) -Kfi*u(8) *u(18) +Kri*u
    (9) ;
f(9) = Kfi*u(8) *u(18) -Kri*u(9) -Kdpgal3*u(9) -Kfd3i80*u
    (16) *u(9) +Kdr3i80*u(17) ;
f(10) = Kipgal4*u(19)-Kdpgal4*u(10) -2*Kfd*u(10) *u(10)
    +2*Krd*u(11) ;
f(11) = Kfd*u(10) *u(10) -Krd*u(11) -Kdpgal4*u(11) ;
f(12) = Kiprep*u(4) -Kdprep*u(12) ;
f(13) = -Kdpgal80*u(13) -Kf80*u(13) +Kr80*u(14) -2*Kfd*u
    (13) *u(13) +2*Krd*u(15) ;
```

```
f(14) = Kipgal80*u(5) + Kf80*u(13) -Kr80*u(14) -2*Kfd*u
   (14)*u(14) +2*Krd*u(16) -Kdpgal80*u(14) ;
f(15) = Kfd*u(13) *u(13) -Krd*u(15) -Kdpgal80*u(15) -Kf80
   *u(15) +Kr80*u(16) ;
f(16) = Kfd*u(14) *u(14) -Krd*u(16) -Kdpgal80*u(16) +Kf80
   *u(15) -Kr80*u(16) -Kfd3i80*u(16) *u(9) +Kdr3i80*u(17)
    ;
f(17) = Kfd3i80*u(16) *u(9) -Kdr3i80*u(17) -0.5*Kdpgal3*u
   (17) ;
f(18) = (Ktr*u(7) *(Gex-(u(18)*4.65*10^-8) )/(Kmtr+Gex+(u
   (18)*4.65*10^-8) +(Atr/Kmtr)*Gex*(u(18)*4.65*10^-8) ))
   -(Kcatgk*u(6) *u(18) /(Kmgk+u(18) ))-Kfi*u(8) *u(18) +
   Kri*u(9) ;
f(19) = Kirgal4 - u(19)*Kdrgal4;
```

Appendix C

Spectral Analysis Methods Comparison Source Code Listing

Listing C.1: fftvsls.m

```
%%
n=10000;
fq1=0.1;
fq2=1.3;
j=0.1;
f1=@(t)sin(2*pi*fq1*t);
f2=@(t)sin(2*pi*fq2*t);
f=@(t)(f1(t)+f2(t));
t=[0:0.01:30];
ts=0:1:30;

%%
s1_acc=zeros(1000,1);
s2_acc=zeros(1000,1);
s3_acc=zeros(1,1025);
```

```
%%
for i=1:n
        tsj=apply_jitter(ts,j,@randn);

        [freq1 s1]=get_spectrum(ts,f(tsj));
        [freq2 s2]=get_spectrum(tsj,f(tsj));

        NFFT = 2^(nextpow2(length(ts))+6); % Next power of 2
            from length of y
        Y = fft(f(tsj),NFFT)/length(ts);
        freq3 = 1/2*linspace(0,1,NFFT/2+1);
        s3=2*abs(Y(1:NFFT/2+1));

        s1_acc=s1_acc+s1;
        s2_acc=s2_acc+s2;
        s3_acc=s3_acc+s3;
end

%%
s1=s1_acc./n; s2=s2_acc./n; s3=s3_acc./n;
s1=s1/max(s1); s2=s2/max(s2); s3=s3/max(s3);

%%
figure();
hold on;
```

```
plot(freq1,s1,'-b','LineWidth',2,'DisplayName','L-S (nominal)
    ');
plot(freq2,s2,'-g','LineWidth',2,'DisplayName','L-S (jitter)'
    );
plot(freq3,s3,'-r','LineWidth',2,'DisplayName','FFT');

axis([0 0.5 0 1]);
xlabel('Frequency (Hz)','FontSize',16);
ylabel('Normalized Magnitude','FontSize',16);
set(gca,'TickDir','out','GridLineStyle',':','Fontsize',16);
%'XGrid','on','YGrid','on',
legend('location','Northeast');

title(['FFTvsLS; n=', num2str(n), ', j=', num2str(j), ', fq1=
    ', num2str(fq1), ', fq2=', num2str(fq2),'; ', datestr(now
    ),30)]);

hold off;
```

Listing C.2: get_spectrum.m

```
function [f s]=get_spectrum(t,y,x)


if(isempty(t))
        t=0:size(y,2)-1;
end
if(nargin<3)
        x=size(t,2);
```

```
end

[freqs spectrum prob]=lomb(t(1:x)',y(1:x)',2000/x,1);
f=freqs;
s=spectrum;
```

Listing C.3: apply_jitter.m

```
function r_tspan = apply_jitter(tspan, amt, r_fct)

r_tspan=tspan;

for i=2:size(tspan,2)
        t_i=tspan(i);
        t_prev=tspan(i-1);
        del_ti=t_i-t_prev;
        j=r_fct();
        shift=del_ti*amt*j;
        while(r_tspan(i-1)>=r_tspan(i)+shift)
                %disp(['  *Point overlap [i=', num2str(i), ',
                    r_tspan(i-1)=', num2str(r_tspan(i-1)), ',
                    r_tspan(i)=', num2str(r_tspan(i)), '
                   del_ti=', num2str(del_ti), ', j=', num2str
                   (j), ', amt=', num2str(amt), ', shift=',
                   num2str(shift), ']']);
                j=r_fct();
                shift=del_ti*amt*j;
        end
```

```
        r_tspan(i)=r_tspan(i)+shift;
end
```