15th European Signal Processing Conference (EUSIPCO 2007), Poznan, Poland, September 3-7, 2007, copyright by EURASIP

ALGORITHMS FOR DOWNSAMPLING NON-UNIFORMLY SAMPLED DATA

Frida Eng, frida@isy.liu.se, *and Fredrik Gustafsson*, fredrik@isy.liu.se

Department of Electrical Engineering, Linköpings universitet, SE-58183 Linköping, Sweden.

ABSTRACT

Decimating a uniformly sampled signal a factor D involves low-pass anti-alias filtering with normalized cut-off frequency $1/D$ followed by picking out every D^{th} sample. Alternatively, decimation can be done in the frequency domain using the fast Fourier transform (FFT) algorithm, after zero-padding the signal and truncating the FFT. We outline three approaches to decimate non-uniformly sampled signals, which are all based on interpolation. The interpolation is done in different domains, and the inter-sample behavior does not need to be known. The first one interpolates the signal to a uniformly sampling, after which standard decimation can be applied. The second one interpolates a continuous-time convolution integral, that implements the anti-alias filter, after which every D^{th} sample can be picked out. The third frequency domain approach computes an approximate Fourier transform, after which truncation and IFFT give the desired result. Simulations indicate that the second approach is particularly useful. A thorough analysis is therefore performed for this case, using the assumption that the non-uniformly distributed sampling instants are generated by a stochastic process.

1. INTRODUCTION

Downsampling is here considered for a non-uniformly sampled signal. Non-uniform sampling appears in many applications, while the cause for non-linear sampling can be classified into one of the following two categories:

Event-based sampling: The sampling is determined by a nuisance event process. One typical example is data traffic in the Internet, where packet arrivals determine the sampling times and the queue length is the signal to be analyzed. Financial data where the stock market valuations are determined by each transaction, is another example.

Uniform sampling in secondary domain: Some angular speed sensors give a pulse each time the shaft has passed a certain angle, so the sampling times depend on angular speed. Also biological signals such as ECGs are naturally sampled in the time domain, but preferably analyzed in another domain (heart rate domain).

A number of other applications and relevant references can be found in, for example, [1].

It should be obvious from the examples above that for most applications, the original non-uniformly sampled signal is sampled much too fast, and that oscillation modes and interesting frequency modes are found at quite low frequencies compared to the inverse mean sampling interval.

In this downsampling problem, we have non-uniform sampling times, t_m, and signal sample values $u(t_m)$, $m = 1,\ldots,M$. The aim is to find the values $z(nT)$, where $z(t)$ is given by filtering of the signal $u(t)$ with the filter $h(t)$, and also the original sampling is faster than necessary, i.e., $T_u = t_M/M \ll T$ ($t_0 = 0$). The aim of the filtering is to remove frequencies above $1/(2T)$, which becomes the effective Nyquist frequency after downsampling.

For the case of uniform sampling, $t_m = mT_u$, two well known solutions exist, see for example, [2]. First, if $T/T_u = D$ is an integer, then (i) $u(mT_u)$ is filtered giving $u_f(mT_u)$, and (ii) $z(nT) = u_f(nDT_u)$ gives the decimated signal.

Further, if $T/T_u = R/S$ is a rational number, then a frequency domain method is known. It is based on (i) zero padding $u(mT_u)$ to length RM, (ii) computing the discrete Fourier transform (DFT), (iii) truncating the DFT a factor S, and finally computing the inverse DFT (IDFT), where the (I)FFT algorithm is used for the (I)DFT.

Resampling and reconstruction are closely connected, since a reconstructed signal can be used to sample at desired time points. The task of reconstruction is well investigated for different setups of non-uniform sampling. A number of iterative solutions have been proposed, e.g., [3, 4, 1], several more are also discussed in [5]. The algorithms are not well-suited for real-time implementations and are based on different assumptions on the sampling times, t_m, such as bounds on the maximum separation or deviation from the nominal value mT_u.

Russel [5] also investigates both uniform and non-uniform resampling thoroughly. Russell argues against the iterative solutions, since they are based on analysis with ideal filters, and no guarantees can be given for approximate solutions. An non-iterative approach is given, which assumes periodic time grids, i.e., the non-uniformity is repeated.

Here, we neither put any constraints on the non-uniform sampling times, nor assumptions on the signal's function class. Instead, we take a more application oriented approach, and aim at good, implementable, resampling procedures. We outline three methods to decimate $u(t_m)$ to $z(nT)$:

- The direct approach, based on interpolating $u(t_m)$ to $u(jT_u)$, where $T_u = t_M/M$ followed by a standard decimation procedure for uniform sampling.
- Convolution interpolation, where a continuous-time low pass filter $h(t)$ is applied to the underlying continuous-time process to give $z(nT) = \int h(nT-\tau)u(\tau)\,d\tau$ and the integrand is interpolated between the available samples $u(t_m)$.
- A frequency domain approach, where the Fourier transform $U(f) = \int u(t)e^{-i2\pi ft}\,dt$ integrand is interpolated.

The first and third algorithm are rather trivial modifications of the time and frequency domain methods for uniformly sampled data, respectively, while the second one is a new truly non-uniform algorithm. We will compare performance of these three. In all three cases, different kinds of interpolation is possible, but we will focus on zero order hold (nearest neighbor) and first order hold (linear interpolation). Of course, which interpolation is best depends on the signal and in particular its inter-sample behavior. Though we prefer to talk about decimation, we want to point out that the theories hold for any type of filter $h(t)$.

A major contribution in this work is a detailed analysis of the algorithms, where we assume Additive Random Sampling, ARS,

$$t_m = t_{m-1} + \tau_m, \tag{1}$$

where τ_m is stochastic additive sampling noise given by the known probability density function $p_\tau(t)$. The theoretical results show that the downsampled signal is unbiased under fairly general conditions and present an equivalent filter that generates $z(t) = \tilde{h} \star u(t)$, where $\tilde{h}$ depends on the designed filter h and the characteristic function of the stochastic distribution.

The rest of the paper is organized as follows. The algorithms are described further in Section 2. The convolutional interpolation gives promising results in the simulations in Section 3, and the Section 4 is dedicated to analysis of this algorithm. Section 5 concludes the paper.

2. INTERPOLATION ALGORITHMS

Time domain interpolation can be used with subsequent filtering. Since LP-filtering is desired, we also propose two other methods that include the filter action directly. The main idea is to perform the interpolation at different levels. The problem at hand is stated as follows:

PROBLEM 1 *The following is given*

- *a sequence of non-uniform sampling times,* t_m, $m = 1, \dots, M$,
- *corresponding signal samples,* $u(t_m)$,
- *a filter impulse response,* $h(t)$, *and*
- *a resampling frequency,* $1/T$.

Also, the desired inter-sampling time, T, *is much larger than the original mean inter-sampling time,*

$$\mu_T \triangleq \mathrm{E}[\tau_m] \approx t_M/M = T_u.$$

Let $\lfloor x \rfloor$ *denote the largest integer smaller than or equal to* x. *Find*

$$\hat{z}(nT), \qquad n = 1, \dots, N, \quad N = \lfloor t_M/T \rfloor \triangleq M/D,$$

such that $|z - \hat{z}|$ *is small, where*

$$z(t) = h \star u(t) = \int h(t-\tau)u(\tau)\,d\tau,$$

is given by convolution of the continuous-time filter $h(t)$ *and signal* $u(t)$.

2.1 Interpolation in Time Domain

It is well described in literature how to interpolate a signal or function in, for instance, the following cases:

- The signal is band-limited, in which case the sinc interpolation kernel gives a reconstruction with no error [6].
- The signal has vanishing derivatives of order $n+1$ and higher, in which case spline interpolation of order n is optimal [7].
- The signal has a bounded second order derivative, in which case the Epanechnikov kernel is the optimal interpolation kernel [8].

The computation burden in the first case is a limiting factor in applications, and for the two other examples, the interpolation is not exact. We consider a simple spline interpolation, followed by filtering and decimation as in Algorithm 1.

Algorithm 1 is optimal only in the unrealistic case where the underlying signal $u(t)$ is piecewise constant between the samples. The error will depend on the relation between the original and the wanted sampling, the larger the ratio M/N the smaller the error. If one assumes a band-limited signal, where all energy of $U(f)$ is restricted to $f < 0.5N/t_M$, then a perfect reconstruction would be possible, after which any type of filtering and sampling can be performed without error. However, this is not a feasible solution in practice, and the band-limited assumption is seldom satisfied for real signals when the sensor is affected by additive noise.

Algorithm 1 Time Domain Interpolation

For Problem 1, with $T_u = t_M/M$, compute

(1) $t_m^j = \arg\min_{t_m} |jT_u - t_m|$

(2) $\hat{u}(jT_u) = u(t_m^j)$

(3) $\hat{z}(kT) = \sum_{j=1}^{M} h_d(kT - jT_u)\hat{u}(jT_u)$

where $h_d(t)$ is a discrete time realization of the impulse response $h(t)$.

REMARK 1 *Algorithm 1 finds* $\hat{u}(jT_u)$ *by nearest-neighbor interpolation, where of course linear interpolation or higher order splines could be used. However, simulations not included showed that this choice does not significantly affect the performance.*

2.2 Interpolation in the Convolution Integral

Filtering of the continuous-time signal, u, yields

$$z(kT) = \int h(kT-\tau)u(\tau)\,d\tau \tag{2}$$

and using Riemann integration we get Algorithm 2. The algorithm will be exact if the integrand, $h(kT-\tau)u(\tau)$, is constant between the sampling points, t_m, for all kT. As before, the error, when this is not the case, is smaller when the ratio M/N is larger.

This algorithm can be further analyzed using the inverse Fourier transform, and the results in [9], which will be done in Section 4. Higher order interpolations of (2) were studied in [10] without finding any benefits.

Algorithm 2 Convolution Interpolation

For Problem 1, compute

(1) $\hat{z}(kT) = \sum_{m=1}^{M} \tau_m h(kT - t_m)u(t_m)$

REMARK 2 *When the filter* $h(t)$ *is causal, the summation is only taken over* m *such that* $t_m < kT$, *and thus Algorithm 2 is ready for on-line use.*

2.3 Interpolation in the Frequency Domain

LP-filtering is given by a multiplication in the frequency domain, and we can form the approximate Fourier transform (AFT), [9], given by Riemann integration of the Fourier transform, to get Algorithm 3. The AFT is formed for $2N$ frequencies to avoid circular convolution. This corresponds to zero-padding for uniform sampling. Then the Inverse DFT computes the estimate. The AFT used in the algorithm is based on Riemann integration of the Fourier transform of $u(t)$, and would be exact whenever $u(t)e^{-i2\pi ft}$ is constant between sampling times, which of course rarely is the case. As for the two previous algorithm, the approximation is less grave for large enough M/N. This paper does not include an investigation of error bounds. More investigations of the AFT were done in [9].

Algorithm 3 Frequency Domain Interpolation

For Problem 1, compute

(1) $f_n = \dfrac{n}{2NT}, \quad n = 0,\ldots,2N-1$

(2) $\hat{U}(f_n) = \sum_{m=1}^{M} \tau_m u(t_m) e^{-i2\pi f_n t_m}, \quad n = 0,\ldots,N$

(3) $\hat{Z}(f_n) = \hat{Z}(f_{2N-n})' = H(f_n)\hat{U}(f_n), \; n = 0,\ldots,N$

(4) $\hat{z}(kT) = \dfrac{1}{2NT} \sum_{n=0}^{2N-1} \hat{Z}(f_n) e^{i2\pi kT f_n}, \quad k = 0,\ldots,N-1.$

Here $\hat{Z}'$ is the complex conjugate of $\hat{Z}$.

3. NUMERIC EVALUATION

We will use the following example to test the performance of these algorithms. The signal consists of three frequencies that are drawn randomly for each test run in order to give a more complete test. Most interesting signals in applications can be described in a similar fashion, but a theoretical analysis would be even more beneficial.

EXAMPLE 1 *A signal with three frequencies, f_j, drawn from a rectangular distribution, Re, is simulated*

$$s(t) = \sin(2\pi f_1 t - 1) + \sin(2\pi f_2 t - 1) + \sin(2\pi f_3 t), \quad (3)$$

$$f_j \in Re(0.01, \frac{1}{2T}), \quad j = 1,2,3. \quad (4)$$

The desired uniform sampling is given by the inter-sampling time $T = 4$ s. The non-uniform sampling is defined by

$$t_m = t_{m-1} + \tau_m, \quad (5)$$

$$\tau_m \in Re(t_l, t_h), \quad (6)$$

and the limits t_l and t_h are varied. In the simulation, N is set to 64 and the number of non-uniform samples are set so that $t_M > NT$ is assured. This is not in exact correspondence with the problem formulation, but assures that the results for different τ_m-distributions are comparable.

The samples are corrupted by additive measurement noise,

$$u(t_m) = s(t_m) + e(t_m), \quad (7)$$

where $e(t_m) \in N(0, \sigma^2)$, $\sigma^2 = 0.1$.

Table 1: RMSE values, λ in (12), for estimation of $\hat{z}(kT)$, in Example 1. The number of runs where respective algorithm finished 1st, 2nd and 3rd, are also shown.

	E[λ]	Std(λ)	1st	2nd	3rd
	Setup in (10a)				
Alg. 1	0.281	0.012	98	258	144
Alg. 2	0.278	0.012	254	195	51
Alg. 3	0.311	0.061	148	47	305
	Setup in (10b)				
Alg. 1	0.338	0.017	9	134	357
Alg. 2	0.325	0.015	175	277	48
Alg. 3	0.330	0.038	316	89	95
	Setup in (10c)				
Alg. 1	0.360	0.018	6	82	412
Alg. 2	0.342	0.015	144	329	27
Alg. 3	0.341	0.032	350	89	61
	Setup in (10d)				
Alg. 1	0.337	0.015	59	133	308
Alg. 2	0.331	0.015	117	285	98
Alg. 3	0.329	0.031	324	82	94

The filter is a second order LP filter of Butterworth type with cut-off frequency $\frac{1}{2T}$, i.e.,

$$h(t) = \sqrt{2}\frac{\pi}{T} e^{-\frac{\pi}{T\sqrt{2}}t} \sin(\frac{\pi}{T\sqrt{2}}t), \qquad t > 0, \quad (8)$$

$$H(s) = \frac{(\pi/T)^2}{s^2 + \sqrt{2}\pi/Ts + (\pi/T)^2}. \quad (9)$$

This setup is used for 500 different realizations of f_j, τ_m and $e(t_m)$.

We will test four different rectangular distributions (6):

$$\tau_m \in Re(0.1, 0.3), \quad \mu_T = 0.2, \quad \sigma_T = 0.06 \quad (10a)$$

$$\tau_m \in Re(0.3, 0.5), \quad \mu_T = 0.4, \quad \sigma_T = 0.06 \quad (10b)$$

$$\tau_m \in Re(0.4, 0.6), \quad \mu_T = 0.5, \quad \sigma_T = 0.06 \quad (10c)$$

$$\tau_m \in Re(0.2, 0.6), \quad \mu_T = 0.4, \quad \sigma_T = 0.12 \quad (10d)$$

and the mean values, μ_T, and standard deviations, σ_T, are shown for reference. For every run we use the algorithms presented in the previous section and compare their results to the exact, continuous-time, result,

$$z(kT) = \int h(kT - \tau)s(\tau)\, d\tau. \quad (11)$$

We calculate the root mean square error, RMSE,

$$\lambda \triangleq \sqrt{\frac{1}{N}\sum_k |z(kT) - \hat{z}(kT)|^2}. \quad (12)$$

The algorithms are ordered according to lowest RMSE, (12), and Table 1 presents the result. The number of first, second and third positions for each algorithm during the 500 runs, are also presented.

A number of conclusions can be drawn from the previous example:

- Comparing a given algorithm for different non-uniform sampling time pdf, Table 1 shows that $p_\tau(t)$, in (10), has a clear effect on the performance.

- Comparing the algorithms for a given sampling time distribution shows that the lowest mean RMSE is no guarantee of best performance at all runs. Algorithm 2 has the lowest E[λ] for setup (10a), but still performs worst in 10% of the cases, and for (10d) Algorithm 3 is number 3 in 20% of the runs, while it has the lowest mean RMSE.
- Usually, Algorithm 3 has the lowest RMSE (1^{st} position), but the spread is more than twice as large (standard deviation of λ), compared to the other two algorithms.
- Algorithms 1 and 2 have similar RMSE statistics, though, of the two, Algorithm 2 performs slightly better in the mean, in all the four tested cases.

In this test we find that Algorithm 3 is often best but Algorithm 2 is almost as good and more stable in its performance. It seems that the specific setup is not as crucial for the performance of Algorithm 2.

REMARK 3 *It is important to note that the performance depends on the setup. For example, Algorithm 2 needs the downsampling factor M/N to be significantly larger than* 1 *for the Riemann approximation to be good. In the examples above it is at least a factor* 10*. As stated before, this is the case for all the algorithms but it still remains to investigate the importance of the setup.*

The algorithms are comparable in performance and complexity. In the following we focus on Algorithm 2, because of its nice analytical properties, its on-line compatibility, and, of course, its slightly better performance results.

4. THEORETIC ANALYSIS OF ALGORITHM 2

Here we study the *a priori* stochastic properties of the estimate, $\hat{z}(kT)$, given by Algorithm 2. For the analytical calculations, we use that the convolution is symmetric, and get

$$\begin{aligned}\hat{z}(kT) &= \sum_{m=1}^{M} \tau_m h(t_m) u(kT - t_m),\\ &= \sum_{m=1}^{M} \tau_m \int H(\eta) e^{i2\pi\eta t_m}\, d\eta \int U(\psi) e^{i2\pi\psi(kT-t_m)}\, d\psi,\\ &= \iint H(\eta) U(\psi) e^{i2\pi\psi kT} \sum_{m=1}^{M} \tau_m e^{-i2\pi(\psi-\eta)t_m}\, d\psi\, d\eta,\\ &= \iint H(\eta) U(\psi) e^{i2\pi\psi kT} W(\psi-\eta; t_1^M)\, d\psi\, d\eta, \quad (13)\end{aligned}$$

with

$$W(f; t_1^M) = \sum_{m=1}^{M} \tau_m e^{-i2\pi f t_m}. \quad (14)$$

Let,

$$\varphi_\tau(f) = E[e^{-i2\pi f\tau}] = \int e^{-i2\pi f\tau} p_\tau(\tau)\, d\tau = \mathscr{F}(p_\tau(t))$$

denote the characteristic function for the sampling noise τ. Here $\mathscr{F}$ is the Fourier transform operator. Then, Theorem 2 in [9] gives

$$E[W(f)] = -\frac{1}{2\pi i}\frac{d\varphi_\tau(f)}{df}\frac{1-\varphi_\tau(f)^M}{1-\varphi_\tau(f)}, \quad (15)$$

and also an expression for the covariance, $\mathrm{Cov}(W(f))$, not repeated here. The expressions are given by straightforward calculations using the fact that the sampling noise sequence τ_m are independent stochastic variables and $t_m = \sum_{k=1}^{m} \tau_k$ in (14). These known properties of $W(f)$ make it possible to find $\mathrm{E}[\hat{z}(kT)]$ and $\mathrm{Var}(\hat{z}(kT))$ for any given characteristic function, $\varphi_\tau(f)$, of the sampling noise, τ_k.

The following Lemma will be useful.

LEMMA 1 (LEMMA 1 IN [9]) *Assume that the continuous-time function $h(t)$ with FT $H(f)$ fulfills the following conditions:*

1. *$h(t)$ and $H(f)$ belongs to the Schwartz class, $\mathscr{S}$.*[1]
2. *The sum $g_M(t) = \sum_{m=1}^{M} p_m(t)$ obeys*

$$\lim_{M\to\infty} \int g_M(t) h(t)\, dt = \int \frac{1}{\mu_T} h(t)\, dt = \frac{1}{\mu_T} H(0), \quad (16)$$

for this $h(t)$.

3. *The initial value is zero, $h(0) = 0$.*

Then, it holds that

$$\lim_{M\to\infty} \int \frac{1-\varphi_\tau(f)^M}{1-\varphi_\tau(f)} H(f)\, df = \frac{1}{\mu_T} H(0). \quad (17)$$

Proof: The proof is conducted using distributions from functional analysis and we refer to [9] for details. ■

Let us study the conditions on $h(t)$ and $H(f)$ given in Lemma 1 a bit more. The restrictions from the Schwartz class could affect the usability of the lemma. However, all smooth functions with compact support (and their Fourier transforms) are in $\mathscr{S}$, which should suffice for most cases. It is not intuitively clear how hard (16) is. Note that, for any ARS case with continuous sampling noise distribution, $p_m(t)$ is approximately a Gaussian for higher m, and we can confirm that, for a large enough fixed t,

$$g_M(t) = \sum_{m=1}^{M} \frac{1}{\sqrt{2\pi m}\sigma_T} e^{-\frac{(t-m\mu_T)^2}{2m\sigma_T^2}} \to \frac{1}{\mu_T}, \quad N \to \infty, \quad (18)$$

with μ_T and σ_T being the mean and the standard deviation of the sampling noise τ, respectively. The integral in (16) can then serve as some kind of mean value approximation, and the edges of $g_N(t)$ will not be crucial. Also, condition 3 further restricts the behavior of $h(t)$ for small t, which will make condition 2 easier to fulfill.

THEOREM 1 *The estimate given by Algorithm 2 can be written as*

$$\hat{z}(kT) = \tilde{h} \star u(kT), \quad (19a)$$

where $\tilde{h}(t)$ is given by

$$\tilde{h}(t) = \mathscr{F}^{-1}(H \star W(f))(t), \quad (19b)$$

with $W(f)$ as in (14)*.*

[1] $h \in \mathscr{S} \Leftrightarrow t^k h^{(l)}(t)$ is bounded, i.e., $h^{(l)}(t) = \mathscr{O}(|t|^{-k})$, for all $k, l \geq 0$.

Furthermore, if the filter $h(t)$ and signal $u(t)$ belong to the Schwartz class, $\mathscr{S}$ [11],

$$\mathrm{E}\,\hat{z}(kT) \to z(kT), \quad \text{if} \quad \sum_{m=1}^{M} p_m(t) \to \frac{1}{\mu_T}, M \to \infty, \tag{19c}$$

$$\mathrm{E}\,\hat{z}(kT) = z(kT), \quad \text{if} \quad \sum_{m=1}^{M} p_m(t) = \frac{1}{\mu_T}, \forall M, \tag{19d}$$

with $\mu_T = E[\tau_m]$, and $p_m(t)$ is the pdf for time t_m.

Proof: First of all, (2) gives

$$z(kT) = \int H(\psi)U(\psi)e^{i2\pi\psi kT}\, d\psi, \tag{20a}$$

and from (13) we get

$$\hat{z}(kT) = \int U(\psi)e^{i2\pi\psi kT} \times \times \underbrace{\int H(\eta)W(\psi-\eta)\, d\eta}_{\triangleq \tilde{H}(\psi)}\, d\psi \tag{20b}$$

which implies that we can identify $\tilde{H}(f)$ as the filter operation on the continuous-time signal $u(t)$, and (19a) follows.

From Lemma 1 and (15) we get

$$\int \mathrm{E}[W(f)]y(f) = \int \mathrm{E}[\tau e^{-i2\pi f\tau}] \frac{1-\varphi_\tau(f)^M}{1-\varphi_\tau(f)} y(f) \to y(0)$$

for any function $y(f)$ fulfilling the properties of the Lemma. This gives

$$\begin{aligned} \mathrm{E}[\hat{z}(kT)] &= \iint H(\eta)U(\psi)e^{i2\pi\psi kT}\, \mathrm{E}[W(\psi-\eta)]\, d\psi\, d\eta, \\ &\to \int H(\psi)U(\psi)e^{i2\pi\psi kT}\, d\psi, \\ &= z(kT), \end{aligned}$$

when $H(f)$ and $U(f)$ behave as requested.

Using the same technique as in the proof of the Lemma, the third property follows. ∎

REMARK 4 *From the investigations in [9] it is clear that $\tilde{H}(f)$, in* (20b)*, is the AFT of the sequence $h(t_m)$, cf., the AFT of $u(t_m)$ in step (2) of Algorithm 3.*

Requiring that both $h(t)$ and $u(t)$ be in the Schwartz class is not, as indicated before, a major restriction. Though, some thought need s to be done for each specific case before applying the theorem.

5. CONCLUSIONS

This work investigated three different algorithms for down-sampling non-uniformly sampled signals, each using interpolation on different levels. Numerical experiments showed that interpolation of the convolution integral presents a good and stable down-sampling alternative, and makes theoretical analysis possible. The algorithm gives asymptotically unbiased estimates for non-causal filters, and the analysis showed the connection between the original filter and the actual filter implemented by the algorithm.

REFERENCES

[1] A. Aldroubi and K. Gröchenig, "Nonuniform sampling and reconstruction in shift-invariant spaces," *SIAM Review*, vol. 43, no. 4, pp. 585—620, 2001.

[2] S. K. Mitra, *Digital Signal Processing*. McGraw-Hill, 1998.

[3] F. J. Beutler, "Error-free recovery of signals from irregularly spaced samples," *SIAM Review*, vol. 8, no. 3, pp. 328–335, July 1966.

[4] F. Marvasti, M. Analoui, and M. Gamshadzahi, "Recovery of signals from nonuniform samples using iterative methods," *IEEE Trans. Signal Process.*, vol. 39, no. 4, pp. 872–878, Apr. 1991.

[5] A. I. Russell, "Regular and irregular signal resampling," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, 2002.

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

[7] M. Unser, "Splines: a perfect fit for signal and image processing," *IEEE Signal Process. Mag.*, vol. 16, no. 6, pp. 22–38, Nov. 1999.

[8] J. Fan and I. Gijbels, *Local Polynomial Modelling and Its Applications*. Chapman & Hall, 1996.

[9] F. Eng, F. Gustafsson, and F. Gunnarsson, "Frequency domain analysis of signals with stochastic sampling times," *IEEE Trans. Signal Process.*, 2007, submitted.

[10] F. Gunnarsson, F. Gustafsson, and F. Gunnarsson, "Frequency analysis using nonuniform sampling with applications to adaptive queue management," in *IEEE Int. Conf. on Acoust., Speech, Signal Processing*, Montreal, Canada, May 2004.

[11] C. Gausquet and P. Witkomski, *Fourier Analysis and Applications*. Springer Verlag, 1999.