Appendix A

Mathematical background

In this appendix, we provide some definitions, equations, and theorems (without proofs), which are used in this thesis. The material was mainly compiled from [Chu92a, Koo93, Dau92, KS91].

A.1 Convolution, delta-function and filtering

The continuous convolution of two functions \( x \) and \( y \) defined on \( \mathbb{R} \) is given by:

\[
(x * y)(t) := \int_{-\infty}^{\infty} x(\tau) y(t - \tau) \, d\tau,
\]

whenever this formula has a meaning. In particular, the convolution product exists for \( x, y \in L^2(\mathbb{R}) \), giving

\[
\|x * y\|_\infty \leq \|x\|_2 \|y\|_2,
\]

and also if \( x \in L^1(\mathbb{R}) \), \( y \in L^p(\mathbb{R}) \). Then

\[
\|x * y\|_p \leq \|x\|_1 \|y\|_p.
\]

Convolution is commutative, i.e., \( x * y = y * x \), and associative, i.e., \( x * (y * z) = (x * y) * z \). Using the theory of distributions, convolution can be defined in another way. These theoretical concepts are also required to prove that the Dirac \( \delta \)-distribution is the unit-element of the convolution operation. We will only state this result:

\[
(\delta \ast x)(t) = (x \ast \delta)(t) = x(t).
\]

The Dirac distribution is not an ordinary function. Often, the Dirac \( \delta \)-distribution is treated as an ordinary function, which is zero everywhere except at \( t = 0 \), and has integral 1.

The discrete convolution between two sequences \( \{a_n\}_{n \in \mathbb{Z}} \) and \( \{b_n\}_{n \in \mathbb{Z}} \) is given by:

\[
(a * x)_n := \sum_{k=-\infty}^{\infty} a_k b_{n-k}.
\]
It is commutative and associative as well. The unit-element is the Kronecker δ-sequence:

\[ \delta_k = \begin{cases} 
1 & k = 0 \\
0 & k \neq 0 
\end{cases} \]

Finally, we define the mixed convolution between a sequence \( \{a_k\}_{k \in \mathbb{Z}} \) and function \( x \) on \( \mathbb{R} \) by:

\[
(a * x)(t) := \sum_{k=-\infty}^{\infty} a_k x(t - k). 
\] (A.3)

The mixed-convolution is neither commutative nor associative.

**Filtering**

The convolution integral (A.1) is used to calculate the response of a linear continuous-time IOM system with impulse response \( h \) to an input signal \( x \). The output signal is given by

\[
y(t) = (x * h)(t). 
\]

A similar relation holds for discrete-time systems with a discrete-time impulse response \( \{h_k\}_{k \in \mathbb{Z}} \):

\[
y_k = (x * h)_k. 
\]

**Convolution and Fourier transform**

The Fourier transform (see next section) turns convolution products into algebraic products in the Fourier domain:

\[
(\mathcal{F}(x * y))(\omega) = X(\omega) Y(\omega). 
\]

This property is often used to facilitate the calculation of convolution integrals.

**A.2 Fourier analysis**

**Fourier transform**

For functions in \( L^1(\mathbb{R}) \), the Fourier transform is defined as:

\[
X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-i\omega t} \, dt. 
\] (A.4)

The Fourier transform of \( x \) is denoted as \( (\mathcal{F}x) \) or \( X \). It has the following properties [Chu92a]:

1. \( X \in L^\infty(\mathbb{R}) \), with \( \|X\|_\infty \leq \|x\|_1 \),
2. \( X \) is uniformly continuous on \( \mathbb{R} \),
3. if the derivative \( x' \) of \( x \) exists and \( x' \in L^1(\mathbb{R}) \), then \( (\mathcal{F}x')(\omega) = i\omega X(\omega) \).
If $X \in L^1(\mathbb{R})$, then the inverse transform is defined as:

$$F^{-1}(X)(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) e^{i\omega t} d\omega,$$  \hspace{1cm} (A.5)

with $x(t) = (F^{-1}X)(t)$ at every point $t$ where $x$ is continuous. If $x \in L^1(\mathbb{R}) \cap L^2(\mathbb{R})$, then $X \in L^2(\mathbb{R})$. The inverse transform exists and lies in $L^2(\mathbb{R})$.

Two functions $x$ and $X$ such that $X = Fx$ and $x = F^{-1}X$ are called a Fourier transform pair, denoted as

$$x(t) \leftrightarrow X(\omega).$$  \hspace{1cm} (A.6)

Assume $f(t) \leftrightarrow F(\omega)$ and $g(t) \leftrightarrow G(\omega)$ are two Fourier transform pairs. Then the following relation holds:

$$\langle f, g \rangle = \frac{1}{2\pi} \langle F, G \rangle \hspace{1cm} \text{Parseval relation.}$$  \hspace{1cm} (A.7)

The Parseval relation is sometimes (weaker) formulated as:

$$\|x\|_2^2 = \frac{1}{2\pi} \|X\|_2^2,$$  \hspace{1cm} (A.8)

which is a special case of the Parseval relation (3.5) given on p. 32.

**Sinc-function and its Fourier transform**

The sinc-function is defined as:

$$\text{sinc}(t) := \begin{cases} \frac{\sin(\pi t)}{\pi t} & x \neq 0 \\ 1 & x = 0 \end{cases}$$  \hspace{1cm} (A.9)

Its Fourier transform is the rect-function:

$$\text{rect}(\omega) := \begin{cases} 1 & |\omega| < \pi \\ 0 & \text{elsewhere} \end{cases}$$  \hspace{1cm} (A.10)

Hence, sinc-function and rect-function form a Fourier transform pair:

$$\text{sinc}(t) \leftrightarrow \text{rect}(\omega).$$

**Fourier series**

A $2\pi$-periodic function $x \in L^2(0, 2\pi)$ can be written in the form of a Fourier series:

$$x(t) = \sum_{k \in \mathbb{Z}} c_k e^{ikt},$$  \hspace{1cm} (A.11)

with $c_k$ being the $k^{\text{th}}$ Fourier coefficient, given by:

$$c_k = \frac{1}{2\pi} \int_{0}^{2\pi} x(t) e^{-ikt} dt.$$  \hspace{1cm} (A.12)
The corresponding Parseval relation is given by

\[ \sum_{k=-\infty}^{\infty} |c_k|^2 = \frac{1}{2\pi} \int_{0}^{2\pi} |x(t)|^2 \, dt. \] (A.13)

The mapping \( L^2(0, 2\pi) \rightarrow l^2 : x(t) \rightarrow c_k \) is also referred to as continuous-to-discrete Fourier transform (CDFT). The CDFT transforms a periodical signal into an infinite-length sequence.

**Infinite-length discrete-time signals**

For sequences in \( l^2(\mathbb{R}) \), the discrete-to-continuous Fourier transform (DCFT) is defined as:

\[ X(\omega) = \sum_{k=-\infty}^{\infty} x_k e^{-ik\omega}, \] (A.14)

with the inverse:

\[ x_k = \frac{1}{2\pi} \int_{0}^{2\pi} X(\omega) e^{ik\omega} \, d\omega. \] (A.15)

The DCFT maps the sequence \( x_k \) onto a \( 2\pi \)-periodic function \( X(\omega) \). If the sequence represents a uniformly sampled, discrete-time signal with sampling interval \( \Delta \), the interval \([-\pi, \pi]\) corresponds to \([-\omega_s/2, \omega_s/2]\) where \( \omega_s = 2\pi/\Delta \).

Note the similarity between DCFT and CDFT. From the mathematical point of view there is no distinction between the two transforms.

**Finite-length discrete-time signals**

The discrete-to-discrete Fourier transform (DDFT) is defined as

\[ X_n = \sum_{k=0}^{N} x_k e^{-ikn}, \] (A.16)

with the inverse:

\[ x_k = \frac{1}{2\pi} \sum_{n=0}^{N} X_n e^{ikn}. \] (A.17)

The DDFT is commonly known as discrete Fourier transform. It can be calculated very efficiently using the Fast Fourier Transform (FFT-algorithm).

**Poisson summation formula**

An important result in Fourier analysis is the Poisson summation formula, which is given here in the most general form:

\[ \sum_{k=-\infty}^{\infty} x(t + kb) e^{-ika} = \frac{1}{|b|} \sum_{n=-\infty}^{\infty} X(\frac{2\pi n + a}{b}) e^{i(2\pi n + a)/b}. \] (A.18)
The Poisson formula is also used in the following form \((a = 0, b = 2\pi)\):

\[
\sum_{k=-\infty}^{\infty} x(t + 2\pi k) = \frac{1}{2\pi} \sum_{n=-\infty}^{\infty} X(n) e^{int}.
\]  

(A.19)

A similar formula holds in the frequency domain \((a = \omega\Delta, b = \Delta, t = 0)\):

\[
\sum_{n=-\infty}^{\infty} X(\omega + n\frac{2\pi}{\Delta}) = \Delta \sum_{k=-\infty}^{\infty} x(k\Delta) e^{-i\omega k\Delta}.
\]  

(A.20)

Another useful form of the Poisson formula is

\[
\sum_{k=-\infty}^{\infty} \langle f(t), g(t-k) \rangle e^{i\omega k} = \sum_{n=-\infty}^{\infty} F(\omega + n2\pi) \overline{G(\omega + n2\pi)}.
\]  

(A.21)

**Translation, modulation and dilation**

The principal operations in wavelet theory are *translation* and *dilation*. We define the following operators [Koo93, p. 19]:

**Translation:** \( T_a x(t) = x(t - a) \)

**Modulation:** \( E_a x(t) = e^{iat} x(t) \)

**Dilation:** \( D_a x(t) = |a|^{-1/2} x(t/a) \) \((a \neq 0)\)

The following inner-product relations hold:

\[
\langle x, T_a y \rangle = \langle T_{-a} x, y \rangle; \quad \langle x, E_a y \rangle = \langle E_{-a} x, y \rangle; \quad \langle x, D_a y \rangle = \langle D_{1/a} x, y \rangle.
\]

Moreover, there are some useful relations with the Fourier transform \(\mathcal{F}\):

\[
\mathcal{F} T_a x = E_{-a} \mathcal{F} x; \quad \mathcal{F} E_a x = T_a \mathcal{F} x; \quad \mathcal{F} D_a x = D_{1/a} \mathcal{F} x.
\]

**A.3 Laplace and Z-transforms**

A convenient tool for control system analysis is the Laplace transform. It is used in favor of the Fourier transform, since it is capable of handling one-sided initial value problems (transients) and signals with exponential growth. The Laplace transform of a signal \(x\) is given by

\[
X(s) = \int_{-\infty}^{\infty} x(t) e^{-st} \, dt,
\]

where \(s\) is the complex Laplace variable, \(s = \sigma + i\omega\). Note that for \(\sigma = 0\) the Laplace transform reduces to the Fourier transform.

The Laplace transform, in general, will not converge for all \(s\). The set \(E \subset \mathbb{C}\) where the integral converges is called the existence region of \(X\). The existence region has the form

\[
E = \{ s \in \mathbb{C} \mid \sigma_1 < \text{Re}(s) < \sigma_2 \},
\]
where \(\sigma_1\) may become minus infinity and/or \(\sigma_2\) plus infinity. In addition, we can define a one-sided Laplace transform for signals that are zero, or not defined, for \(t < 0\). This transform is given by

\[
X(s) = \int_0^{\infty} x(t) e^{-st} dt,
\]

and its convergence region is \(E = \{s \in \mathbb{C} \mid \text{Re}(s) > \sigma_1\}\). For signals that are zero for \(t < 0\) both transforms are identical.

The inverse Laplace transform is:

\[
x(t) = \frac{1}{2\pi i} \int_{\sigma-i\infty}^{\sigma+i\infty} X(s) e^{st} ds,
\]

where \(\sigma\) is chosen inside the existence region. Evaluation of this integral requires complex function theory. It can be often avoided: the inverse Laplace transform can also be obtained by writing \(X(s)\) as a sum of simple terms whose inverse transforms are known. This procedure is called partial fraction expansion. An example of this expansion is

\[
X(s) = \frac{s - 1}{(s + 1)^2(s - 2)} = \frac{-1/9}{s + 1} + \frac{2/3}{(s + 1)^2} + \frac{1/9}{s - 2}.
\]

Assuming one-sided signals defined for \(t \geq 0\), the inverses of the three terms at the right-hand side can be found in a table of Laplace transforms, giving the inverse:

\[
x(t) = -1/9 e^{-t} + 2/3 t e^{-t} + 1/9 e^{2t}.
\]

The discrete-time equivalent of the Laplace transform is the Z-transform

\[
X(z) = \sum_{k=-\infty}^{\infty} x_k z^{-k}.
\]

Here \(z\) is a complex variable: \(z = \rho e^{j\omega \Delta}\), and \(\Delta\) is the sampling interval. The existence region is

\[
E = \{z \in \mathbb{C} : \rho_1 < |z| < \rho_2\},
\]

where \(\rho_1\) may become zero and \(\rho_2\) infinity. For \(\rho = 1\), the Z-transform reduces to the DCFT. The one-sided variant is given by

\[
X(z) = \sum_{k=0}^{\infty} x_k z^{-k},
\]

with existence region \(E = \{z \in \mathbb{C} : |z| > \rho_1\}\). The inverse transform is

\[
x_k = \frac{1}{2\pi i} \oint_{\Gamma} X(z) z^k \frac{dz}{z},
\]

where \(\Gamma\) is a contour within the existence region. In most situations, the inverse can be found by partial fraction expansion or by expanding \(X(z)\) in a power series.

The reader is referred to [KS91] for a more detailed discussions of Laplace and Z-transforms.
Appendix B

Mathematica program for MSDS analysis

The following piece of Mathematica code has been used to calculate the transfer function $K(z)$ (6.15) on p. 124. Though $K[z]$ is the final result, little overhead is required to split the numerator and denominator and to write them in MATLAB format on a file.

```
rules={x^m_.^n_.->x^(m*n)}
$Post = (# /. rules) &;

W = Exp[-I * 2 * Pi/M];
DD[H_, z_, M_] := 1/M * Sum[H /. {z -> z^(1/M)*W^k}, {k, 0, M-1}];
UU[H_, z_, M_] := H /. {z -> z^M};
zoh[z_] := Sum[z^i, {i, -M+1, 0}]/M;

(*---------------------------------------------------------------*)
(* input section *)

M = 2
Pname = "z"; (* Use lower case letters: n,z,f,b *)
Qname = "b";
G[z_] := (a*z + b)/(z^2 + c*z + d);
H[z_] := 1;

(*---------------------------------------------------------------*)

Switch[Pname,
   "n", P[z_] := 1,
   "z", P[z_] := zoh[z],
   "f", P[z_] := zoh[z]^2,
   "b", P[z_] := zoh[z]^3 * (1 + z)/(2*z)];

Switch[Qname,
   "n", Q[z_] := M
   "z", Q[z_] := zoh[z]^M
   "f", Q[z_] := zoh[z]^2 * M,
   "b", Q[z_] := zoh[z]^3 *(1 + z)/(2*z) * M];

K[z_] := (H[z^M]*Q[z]*G[z])/(1+H[z^M]*UU[DD[P[z]*Q[z]*G[z],z,M],z,M]);

(*---------------------------------------------------------------*)
```
(* output *)

s = "==========================================================="
ss = "-----------------------------------------------------------"
s >>> outfile
G[z] >>> outfile
H[z] >>> outfile
M >>> outfile
StringJoin["Filter type: ", Pname, Qname] >>> outfile
ss >>> outfile

n[z_] := Expand[Numerator[Factor[K[z]]]];
InputForm[Reverse[CoefficientList[n[z], z]]] >>> outfile
n[z_] := Expand[Denominator[Factor[K[z]]]];
InputForm[Reverse[CoefficientList[n[z], z]]] >>> outfile
r >>> outfile
Appendix C

The sampling toolbox

The SAMPLING TOOLBOX is a group of MATLAB functions for calculating the filters and simulating the general sub-sampling scheme of Chapter 5.

The toolbox can be distributed freely under the GNU-publishing licence. It can be obtained by anonymous ftp at the following URL:


The toolbox consists of the following functions

% splines and spline-wavelets
[x,t] = BSPLINE(m,sf) sampled b-spline of degree m
[x,p] = BSPLWAV(n,m) b-spline wavelet
[x,p] = CSPLINE(n,m) cardinal spline to b-spline
[x,q] = CSPLWAV(n,m,prec) cardinal b-spline wavelet
[x,p] = DCSPLINE(n,m) dual spline to cardinal spline
[x,p] = DSPLINE(n,m) dual spline to b-spline
[x,q] = DSPLWAV(n,m) dual to b-spline wavelet
[x,q] = OSPLINE(n,m) orthogonal spline to b-spline
[x,q] = OSPLWAV(n,m) orthogonal b-spline wavelet

% sinc-functions and filters
x = DSINC(t,fc) discrete sinc function (= ideal low-pass in periodic domain)
x = DSINCF(t,dsf) discrete sinc filter (= ideal low-pass in periodic domain)
x = SINC(t,fc) sinc function (= ideal low-pass)
x = SINCF(t,dsf) sinc filter with relative cut-off frequency

% Daubechies filters
h = DAUBFILT(order,dsf) direct wavelet FIR filters

% up/down sampling
y = DSAMP(x,dsf) down-sampling of array x by factor dsf
y = MDSAMP(x,dsf) down-sampling from maximum value
y = SDSAMP(x,dsf) symmetry preserving down-sampling
y = SUSAMP(x,usf) symmetry preserving up-sampling
y = USAMP(x,usf) up-sampling

% special convolution
z = CCONV(x,y) cyclic convolution via FFT and IFFT
y = CONVINV(x,prec,pad) convolution inverse
y = CONVROOT(x) convolution inverse square root
y = CONVROOT(x,prec,pad) convolution square root
z = FCONV(x,y) convolution and truncation (symmetrical filter y)
y = TRUNCFILT(x,frac) truncation of a long filter
y = ZEROPAD(x,n) symmetrical zero padding

% misc
k = ISSYM(x,prec) symmetry detection of an (odd) sequence
x = ITFILT(h,M) iterates a filter from sequence h using
    the two-scale relation

% general subsampling scheme
y = GENSCHEM(x,p,q,M) general subsampling scheme

For example, the output of the general sub-sampling scheme of Figure 5.1 on p. 87 is calculated
by the following MATLAB statements (assuming input signal x):

M = 2;
n = 2;
p = bspline(n,M);
q = dspline(n,M);
y = fconv(x,M);
ystar = usamp(dsamp(y,M),M);
z = fconv(ystar,fliplr(q))

The function genschem implements the general subsampling scheme with filters p and q and
down-sampling factor M.