

A Network Model of Analogue Computation over Metric Algebras ^{*}

J.V. Tucker¹ and J.I. Zucker^{2, **}

¹ Department of Computer Science, University of Wales Swansea,
Singleton Park, Swansea, SA2 8PP, UK

`J.V.Tucker@swansea.ac.uk`

² Department of Computing & Software, McMaster University,
Hamilton, Ontario L8S 4K1, Canada

`zucker@mcmaster.ca`

Abstract. We define a general concept of a network of analogue modules connected by channels, processing data from a metric space A , and operating with respect to a global continuous clock \mathbb{T} . The inputs and outputs of the network are continuous streams $u : \mathbb{T} \rightarrow A$, and the input-output behaviour of the network with system parameters from A is modelled by a function $\Phi : \mathcal{C}[\mathbb{T}, A]^p \times A^r \rightarrow \mathcal{C}[\mathbb{T}, A]^q$ ($p, q > 0, r \geq 0$), where $\mathcal{C}[\mathbb{T}, A]$ is the set of all continuous streams equipped with the compact-open topology. We give an equational specification of the network, and a semantics which involves solving a fixed point equation over $\mathcal{C}[\mathbb{T}, A]$ using a contraction principle. We analyse a case study involving a mechanical system. Finally, we introduce a custom-made concrete computation theory over $\mathcal{C}[\mathbb{T}, A]$ and show that if the modules are concretely computable then so is the function Φ .

1 Introduction

Let us take analogue computation to be computation by the application of experimental procedures, notably measurement, to physical, chemical or biological systems. Analogue computation is based on continuous data, such as real numbers and data streams. The systems are networks of components or modules that operate in continuous time.

Historically, in analogue computation as conceived by Kelvin [TT80] and Bush [Bus31], data are represented by measurable physical quantities such as length, voltage, etc., processed by networks of mechanical or electrical components. Currently, analogue computation can involve a much wider range of technologies, inspired, for example, by neural networks and cellular automata.

Digital computation, on the other hand, is fundamentally computation by algorithms on discrete data in discrete time. Starting in the 1930s, classical computability theory has matured into a comprehensive and mathematically deep *theory of digital computation*. Turing computability and its equivalents have become the standard for what we mean by computation. The subject continues to

^{*} To appear in the Proceedings of *CiE 2005: Computation in Europe*, Amsterdam, June 2005, ed. S.B. Cooper, B. Loewe and L. Torenvliet, LNCS, Springer-Verlag.

^{**} The research of the second author was supported by a grant from the Natural Sciences and Engineering Research Council of Canada

develop in new directions [Gri99]. Of particular relevance is Computable Analysis, where it is applied to computable functions on real numbers, Banach spaces, and, more generally, metric and topological spaces.

The *theory of analogue computation* is less developed. The general purpose analog computer (GPAC) was introduced by Shannon [Sha41] to model Bush’s Differential Analyzer. Shannon discovered that a function can be generated by a GPAC if, and only if, it is differentially algebraic, but his proof was incomplete. Marian Pour-El [PE74] gave a characterisation of the analogue computable functions, focusing on the classic analogue systems built from adders, scalar multipliers and integrators. This yielded a new proof of Shannon’s equivalence and a proof that these analogue models do not compute all algorithmically (or “digitally”) computable functions on the reals.

Cristopher Moore [Moo96] defined a system of schemes rather like Kleene’s [Kle52], but with primitive recursion replaced by integration. Félix Costa and his colleagues [GC03,MC04] have presented improved models extending GPAC.

We present two questions related to analogue technology:

1. What characteristics of data, physical components, transmissions, and system architecture, make up a suitable technology for analogue computation?
2. Given a technology that builds analogue systems from components, do these systems produce, by measurements, the same functions as those algorithmically computed?

Thanks to the work of Shannon, Pour-El, Moore and Costa, we have one possible precise formulation of question 1, and negative answer to question 2. Their models are based on the traditional components of analogue computing up to the 1960s (adders, integrators, etc.). However, even for the case of traditional analogue technologies, the conceptual basis is not sufficiently clear to answer (even) the first question fully.

We begin, in Section 2, with a definition of an analogue network, with modules connected by channels, processing data from a metric space A , with a global continuous clock \mathbb{T} modelled by the set of non-negative reals. Let $\mathcal{C}[\mathbb{T}, A]$ be the set of all continuous streams $u : \mathbb{T} \rightarrow A$ with the compact-open topology. The input-output behaviour of a network N with p input channels, q output channels and r parameters from A is modelled by a function $\Phi : \mathcal{C}[\mathbb{T}, A]^p \times A^r \rightarrow \mathcal{C}[\mathbb{T}, A]^q$. The module functions must satisfy an important physically motivated condition: *causality*. We give an equational specification for N .

In Section 3 we give a semantics for the equational specification of a network satisfying causality. This involves solving a fixed point equation over $\mathcal{C}[\mathbb{T}, A]$ using a custom-made *contraction principle*, based on the fact that $\mathcal{C}[\mathbb{T}, A]$ can be locally approximated by metric spaces. This extends the well-known Banach fixed point theorem for metric spaces [Eng89]. We also derive continuity of Φ , assuming continuity of the module functions. This gives a mathematical model of *computation by measurements on an analogue system*.

In Section 4 we analyse in detail a case study of analogue computation with a mechanical system in which data are represented by displacement, velocity and acceleration.

In Section 5 we compare analogue and digital computation. For this we introduce a custom-made concrete (digital) computation theory over $\mathcal{C}[\mathbb{T}, A]$. This is an extension to the non-metric space $\mathcal{C}[\mathbb{T}, A]$ of the theory of concrete computations on metric algebras [TZ04]. We prove a soundness theorem for analogue, relative to concrete, computation:

Theorem. *If the functions defined by the components of an analogue network are concretely computable, then so is the function defined by the whole network.*

Settling a converse result, i.e. completeness of analogue with respect to digital computation, would be of great importance.

We have studied computation on discrete time streams in [TZ94], and networks that process discrete time streams in [TT91].

2 Analogue networks

An *analogue network* N consists of a number of *modules* and *channels* computing and communicating with data from a topological algebra A .

2.1 Data and time. Assume we are working with data from a *complete metric space* (A, d) . The network operates in continuous time \mathbb{T} , modelled by the non-negative reals with its usual topology. The channels carry signals in the form of *continuous streams* of data from A , represented as continuous functions $u : \mathbb{T} \rightarrow A$. Let $\mathcal{C}[\mathbb{T}, A]$ be the set of continuous streams on A , with the *compact-open topology* [Eng89].

2.2 Modules. A *module* M has finitely many *input channels*, one *output channel*, and locations for some *parameters*. Associated with M is a function $F_M : \mathcal{C}[\mathbb{T}, A]^{k_M} \times A^{l_M} \rightarrow \mathcal{C}[\mathbb{T}, A]$, with $k_M > 0$ *input streams*, $l_M \geq 0$ *parameters* and one *output stream*. We put $F_M(\mathbf{u}, \mathbf{c}) = v$, where $\mathbf{u} = (u_1, \dots, u_{k_M}) \in \mathcal{C}[\mathbb{T}, A]^{k_M}$ and $\mathbf{c} = (c_1, \dots, c_{l_M}) \in A^{l_M}$.

Examples 2.2.1. Typical module operations (assuming $A = \mathbb{R}$) are the classical analogue processing units: (a) pointwise addition of two streams, (b) pointwise multiplication of a stream by a constant (“scalar”), (c) integration. There are parameters in (a) and (c), namely the scalar multiplier in (a), and the constant of integration in (c).

We will assume a *causality* property of the module functions, which states that the output is “causally” related to the inputs, in the sense that the output at any time depends only on the inputs up to that time. Precisely:

(**Caus**): For $\mathbf{u}_1, \mathbf{u}_2 \in \mathcal{C}[\mathbb{T}, A]^{k_M}$, $\mathbf{c} \in A^{l_M}$ and $t \geq 0$:

$$\mathbf{u}_1 \upharpoonright_{[0,t]} = \mathbf{u}_2 \upharpoonright_{[0,t]} \implies F_M(\mathbf{u}_1, \mathbf{c})(t) = F_M(\mathbf{u}_2, \mathbf{c})(t).$$

Note that this condition depends on an assumption of *instantaneous response* of the modules. All the common module operations (including those listed in 2.2.1) satisfy (**Caus**).

2.3 Network architecture. Consider now (Figure 1) a network N with m modules M_1, \dots, M_m and m channels $\alpha_1, \dots, \alpha_m$. Each module M_i ($i = 1, \dots, m$) has some *input channels* $\alpha_{i_1}, \dots, \alpha_{i_{k_i}}$ ($k_i > 0$) (which are the outputs of modules $M_{i_1}, \dots, M_{i_{k_i}}$ respectively), some (*local*) *parameter locations* $c_{i_1}, \dots, c_{i_{l_i}}$ ($l_i \geq 0$) and one *output channel* α_i . It computes the function $F_i = F_{M_i} : \mathcal{C}[\mathbb{T}, A]^{k_i} \times A^{l_i} \rightarrow \mathcal{C}[\mathbb{T}, A]$.

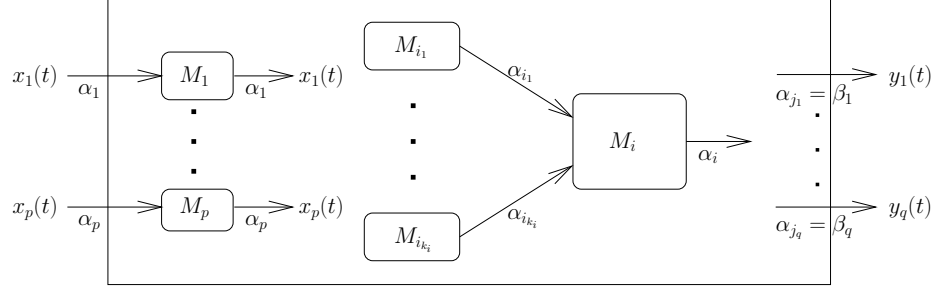


Figure 1. A network

The network N itself has p *input channels* and q *output channels* ($p, q \leq m$). We assume (for notational convenience) that the first p modules M_1, \dots, M_p are the identity module $M_{\mathbf{I}}$, and the p network input channels $\alpha_1, \dots, \alpha_p$ are both the input and output channels for $M_{\mathbf{I}}$. The remaining (non-trivial) modules of the network are M_{p+1}, \dots, M_m . For $i = 1, \dots, m$, the channel α_i is the output channel for module M_i . The q network output channels are β_1, \dots, β_q , where (say) $\beta_i = \alpha_{j_i}$ for $i = 1, \dots, q$.

There are also locations for *global* or *network parameters* $\mathbf{c} = (c_1, \dots, c_r)$ ($r \geq 0$), which include the local parameters of all the modules in N . For each global parameter c_i and module M_j , it is specified which of the local parameters of M_j are to be identified with c_i .

We make an assumption of *input determinacy*:

(InDet): *There is a well-determined value for the stream on each input channel at all times.*

2.4 Network operation: the model. Under the assumptions **(InDet)** and **(Caus)**, we want to prove a *network determinacy* condition:

(NetDet): *For certain input streams and parameter values, there is a well-determined value for the stream on each channel at all times.*

This means that, at least for a certain set $U \subseteq \mathcal{C}[\mathbb{T}, A]^p \times A^r$ of inputs and parameters, there is a well-determined tuple of total functions $u_i : \mathbb{T} \rightarrow A$ ($i = 1, \dots, m$) that describes the data on every channel α_i .

Assuming (*NetDet*), there is associated with each module M_i ($i = 1, \dots, m$) a function $\Phi_i : \mathcal{C}[\mathbb{T}, A]^p \times A^r \rightarrow \mathcal{C}[\mathbb{T}, A]$ where $\Phi_i(\mathbf{x}, \mathbf{c}) = u_i$ for $(\mathbf{x}, \mathbf{c}) \in U$. From these module functions follows the existence of the *network function*

$$\begin{aligned} \Phi^N : \mathcal{C}[\mathbb{T}, A]^p \times A^r &\rightarrow \mathcal{C}[\mathbb{T}, A]^q, \\ \Phi^N(\mathbf{x}, \mathbf{c}) &= (\Phi_{j_1}(\mathbf{x}, \mathbf{c}), \dots, \Phi_{j_q}(\mathbf{x}, \mathbf{c})) \quad \text{for } (\mathbf{x}, \mathbf{c}) \in U. \end{aligned} \quad (2.1)$$

2.5 Network operation: algebraic specification. Given the above assumptions, we can specify the model by the following *system equations*:

$$u_i(t) = F_i(u_{i1}, \dots, u_{ik_i}, c_{i1}, \dots, c_{il_i})(t) \quad (i = 1, \dots, m, t \geq 0) \quad (2.2a)$$

$$u_i(t) = x_i(t) \quad (i = 1, \dots, p, t \geq 0), \quad (2.2b)$$

In the next section we will derive the existence and uniqueness of a solution of this specification as a fixed point of a certain function.

3 Solving network equations; Fixed point semantics

We are looking for an m -tuple of channel functions satisfying the equational specifications (2.2). First, we define some general concepts and give some results concerning stream spaces and stream transformations. Recall that (A, d) is a complete metric space.

3.1 Stream spaces and stream transformations. Let $0 \leq a < b$, and let $\mathcal{C}[[a, b], A]$ be the set of continuous functions from $[a, b]$ to A . For $u, v \in \mathcal{C}[[a, b], A]$ (or $u, v \in \mathcal{C}[\mathbb{T}, A]$), define

$$d_{a,b}(u, v) =_{df} \sup \{d(u(t), v(t)) \mid t \in [a, b]\}.$$

This makes $\mathcal{C}[[a, b], A]$ a complete metric space, with the *uniform convergence* topology [Eng89, §2.6]. The product space $\mathcal{C}[[a, b], A]^m$ ($m > 0$) has the metric

$$d_{a,b}^m(\mathbf{u}, \mathbf{v}) = \left(\sum_{i=1}^m (d_k(u_i, v_i))^p \right)^{\frac{1}{p}} \quad (3.1)$$

(where $\mathbf{u} = (u_1, \dots, u_m)$ and $\mathbf{v} = (v_1, \dots, v_m)$) for some fixed p ($1 \leq p \leq \infty$). We will sometimes drop the superscript ‘ m ’ from $d_{a,b}^m$. We also write d_k for $d_{0,k}$ ($k = 1, 2, \dots$).

The *stream space* $\mathcal{C}[\mathbb{T}, A]$ is, in general, not a metric space, and $d_{a,b}$ is only a pseudometric on $\mathcal{C}[\mathbb{T}, A]$. Nevertheless we can define a notion of convergence in $\mathcal{C}[\mathbb{T}, A]$ as follows. A sequence (u_0, u_1, u_2, \dots) of elements of $\mathcal{C}[\mathbb{T}, A]$ is said to *converge locally uniformly* to the limit $u \in \mathcal{C}[\mathbb{T}, A]$ if for all k there exists N such that for all $n \geq N$, $d_k(u_n, u_N) \leq 2^{-k}$. The space $\mathcal{C}[\mathbb{T}, A]$ is given the *compact open topology* [Eng89, §3.4]. This is equivalent to the *topology of local uniform*

convergence, which can be characterised as follows. Given a set $X \subseteq \mathcal{C}[\mathbb{T}, A]$ and a point $u \in \mathcal{C}[\mathbb{T}, A]$, u is in the *closure* of X if, and only if, there is a sequence of elements of X which converges locally uniformly to u .

This topology on $\mathcal{C}[\mathbb{T}, A]$ can also be characterised as the *inverse limit* [Eng89] of the family of topological spaces $\mathcal{C}[[0, k], A]$ ($k = 0, 1, 2, \dots$) with mappings $\pi_k : \mathcal{C}[[0, k+1], A] \rightarrow \mathcal{C}[[0, k], A]$ defined by $\pi_k(u) = u \upharpoonright_k$.

The space $\mathcal{C}[\mathbb{T}, A]$ is *complete* in the following sense. We must first define:

Definition 3.1.1 (Locally uniform Cauchy sequence). A sequence (u_0, u_1, u_2, \dots) of elements of $\mathcal{C}[\mathbb{T}, A]$ is *locally uniform Cauchy* if $\forall k \exists N \forall m, n \geq N : d_k(u_m, u_n) \leq 2^{-k}$.

Lemma 3.1.2 (Completeness of $\mathcal{C}[\mathbb{T}, A]$). A *locally uniform Cauchy sequence* in $\mathcal{C}[\mathbb{T}, A]$ converges locally uniformly to a limit.

We are interested in stream transformations $f : \mathcal{C}[\mathbb{T}, A]^m \rightarrow \mathcal{C}[\mathbb{T}, A]^m$.

Definition 3.1.3 (Contracting stream transformations). Let $0 < \kappa < 1$ and $\tau > 0$. A stream transformation $f : \mathcal{C}[\mathbb{T}, A]^m \rightarrow \mathcal{C}[\mathbb{T}, A]^m$ is *contracting w.r.t.* (κ, τ) , or in $\mathbf{Contr}(\kappa, \tau)$, if for all $T \geq 0$ and all $\mathbf{u}, \mathbf{v} \in \mathcal{C}[\mathbb{T}, A]^m$:

$$d_{T, T+\tau}(f(\mathbf{u}), f(\mathbf{v})) \leq \kappa \cdot d_{T, T+\tau}(\mathbf{u}, \mathbf{v}).$$

Lemma 3.1.4. Suppose f satisfies (*Caus*). If $f \in \mathbf{Contr}(\kappa, \tau)$ for some $\tau > 0$, then $f \in \mathbf{Contr}(\kappa, \tau')$ for all $\tau' > 0$.

Remark 3.1.5. Hence if $f \in \mathbf{Contr}(\kappa, \tau)$, we can choose τ freely. We will henceforth write $\mathbf{Contr}(\kappa)$ instead of $\mathbf{Contr}(\kappa, \tau)$, and generally take $\tau = 1$.

Theorem 1 (Fixed point of contracting stream transformation).

Suppose $f \in \mathbf{Contr}(\kappa)$ for some $\kappa < 1$. Then f has a unique fixed point, i.e., there is a unique $\mathbf{u} \in \mathcal{C}[\mathbb{T}, A]^m$ such that $f(\mathbf{u}) = \mathbf{u}$.

Proof. Uniqueness is an easy exercise. We prove existence by constructing a fixed point \mathbf{u} of f as a limit of a locally uniformly convergent Cauchy sequence of stream tuples:

$$\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \dots \tag{3.2}$$

Define \mathbf{u}_0 arbitrarily, and $\mathbf{u}_{n+1} = f(\mathbf{u}_n)$. Then for all k, n , by induction on n :

$$d_k(\mathbf{u}_{n+1}, \mathbf{u}_n) \leq \kappa^n d_k(\mathbf{u}_1, \mathbf{u}_0).$$

The sequence (3.2) can then be seen to be a locally uniform Cauchy sequence, by choosing N (for a given k , in Definition 3.1.1) such that

$$\kappa^N < \frac{2^{-k}}{d_k(\mathbf{u}_1, \mathbf{u}_0)}.$$

Thus by Lemma 3.1.2, the sequence (3.2) converges locally uniformly to a limit \mathbf{u} . Hence, also, the sequence

$$f(\mathbf{u}_0), f(\mathbf{u}_1), f(\mathbf{u}_2), \dots \tag{3.3}$$

converges locally uniformly to $f(\mathbf{u})$, since by the contraction property of f ,

$$d(f(\mathbf{u}_n), f(\mathbf{u})) \leq \kappa \cdot d(\mathbf{u}_n, \mathbf{u}).$$

Since (3.3) is the sequence (3.2) shifted by 1, it also converges to \mathbf{u} , and so $f(\mathbf{u}) = \mathbf{u}$. \square

In Section 5, where we consider the *computability* of the fixed point \mathbf{u} as a function of the inputs (\mathbf{x}, \mathbf{c}) , we will need a stronger property of the sequence (3.2) than local uniform convergence, namely *effective* local uniform convergence.

We turn to apply the above theory to the network N .

3.2 Network function. Recall the network function Φ^N (2.1) and the specifications (2.2). Notice next that a stream tuple (u_1, \dots, u_m) satisfying the specifications (2.2) can be characterised as a *fixed point* of the function

$$\begin{aligned} \Psi_{\mathbf{c}}^N : \mathcal{C}[\mathbb{T}, A]^m &\rightarrow \mathcal{C}[\mathbb{T}, A]^m \\ \text{defined by } \Psi_{\mathbf{c}}^N(u_1, \dots, u_m) &= (F_1(\mathbf{u}_1, \mathbf{c}_1), \dots, F_m(\mathbf{u}_m, \mathbf{c}_m)) \end{aligned} \quad (3.4)$$

(where, on the r.h.s., $\mathbf{u}_i, \mathbf{c}_i$ are the lists of input streams and local parameters associated with F_i) subject to the constraint (2.2b). Now by equation (2.2b), the first p components (u_1, \dots, u_p) of the tuple (u_1, \dots, u_m) on the left hand side are identical to \mathbf{x} . Similarly, on the right hand side, for $i = 1, \dots, p$, F_i is the identity function, with argument $u_i = x_i$, and so (3.4) can be rewritten as

$$\Psi_{\mathbf{c}}^N(\mathbf{x}, u_{p+1}, \dots, u_m) = (\mathbf{x}, F_{p+1}(\mathbf{u}_{p+1}, \mathbf{c}_{p+1}), \dots, F_m(\mathbf{u}_m, \mathbf{c}_m)). \quad (3.5)$$

Therefore Ψ^N can be reformulated as a function only of the *non-input streams* $\mathbf{u} = (u_{p+1}, \dots, u_m)$, with the *input streams* \mathbf{x} as further parameters, thus:

$$\begin{aligned} \Psi_{\mathbf{c}, \mathbf{x}}^N : \mathcal{C}[\mathbb{T}, A]^{m-p} &\rightarrow \mathcal{C}[\mathbb{T}, A]^{m-p} \\ \Psi_{\mathbf{c}, \mathbf{x}}^N(\mathbf{u}) &=_{df} \Psi_{\mathbf{c}}^N(\mathbf{x}, \mathbf{u}). \end{aligned} \quad (3.6)$$

So a *fixed point* for $\Psi_{\mathbf{c}, \mathbf{x}}^N$ will be a *solution* to (2.2). Thus the basic questions are:

- Under what conditions does $\Psi_{\mathbf{c}, \mathbf{x}}^N$ have a fixed point?
- Under what conditions is it unique?

We will give at least a partial solution to this, namely a sufficient condition for a fixed point, by applying the theory of contracting stream transformations developed above.

3.3 Solution of fixed point equation. Recall Def. 3.1.3 and Remark 3.1.5.

Definition 3.3.1 (Contracting condition for network). Given $\mathbf{c} \in A^r$, $\mathbf{x} \in \mathcal{C}[\mathbb{T}, A]^p$ and $0 < \kappa < 1$, the network N satisfies $\mathbf{Contr}_{\mathbf{c}, \mathbf{x}}(\kappa)$ if the stream transformation $\Psi_{\mathbf{c}, \mathbf{x}}^N$ is in $\mathbf{Contr}(\kappa)$. It is *contracting at* (\mathbf{c}, \mathbf{x}) if it satisfies $\mathbf{Contr}_{\mathbf{c}, \mathbf{x}}(\kappa)$ for some $\kappa < 1$.

Theorem 2.

- (a) Suppose for all $(\mathbf{x}, \mathbf{c}) \in U \subseteq \mathcal{C}[\mathbb{T}, A]^p \times A^r$, there exists $\kappa < 1$ such that the network N satisfies $\mathbf{Contr}_{\mathbf{c}, \mathbf{x}}(\kappa)$. Then for all $(\mathbf{x}, \mathbf{c}) \in U$ there is a unique $\mathbf{u} = (u_1, \dots, u_m) \in \mathcal{C}[\mathbb{T}, A]^m$ satisfying (2.2). It is given by specifying $u_i = x_i$ for $i = 1, \dots, p$, and $\mathbf{u} = (u_{p+1}, \dots, u_m)$ as the unique fixed point of the function $\Psi_{\mathbf{c}, \mathbf{x}}^N$ defined by equations (3.5) and (3.6). This defines the network function Φ^N as in (2.1), with $\Phi^N(\mathbf{x}, \mathbf{c}) = \mathbf{u}$ for all $(\mathbf{x}, \mathbf{c}) \in U$.
- (b) If, in addition, the module functions are continuous, then Φ^N is continuous at all points in U at which κ can be defined continuously.

Part (a) is immediate from Theorem 1. We omit the proof of (b).

4 A case study

We apply the theory of Section 3 to an example from a standard text [Hyn70].

4.1 The physical system. (See Figure 2.) A mass M is suspended by a spring with stiffness K and damping coefficient D . A force f (varying with time t) is applied to M . We want to compute its displacement x as a function of t .

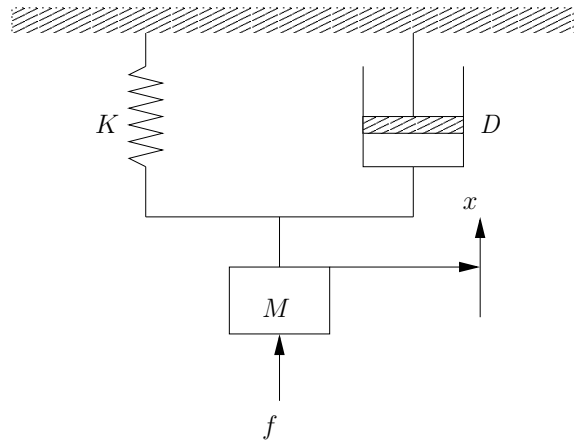


Figure 2. Case study: the physical system

4.2 Equational specification. Three forces act on the mass: the external force f , the spring force $-Kx$, and the damping force $-Ddx/dt$. By Newton's second law of motion, $Ma + Dv + Kx = f$, where $v = dx/dt$ is the velocity, and $a = dv/dt$ the acceleration.

4.3 Network. The analogue network N for this system is shown in Figure 3. It is simplified from the one in [Hyn70], by combining each scalar multiplier with the preceding or following module. There is also an extra “identity” module M_1 for the input stream f .

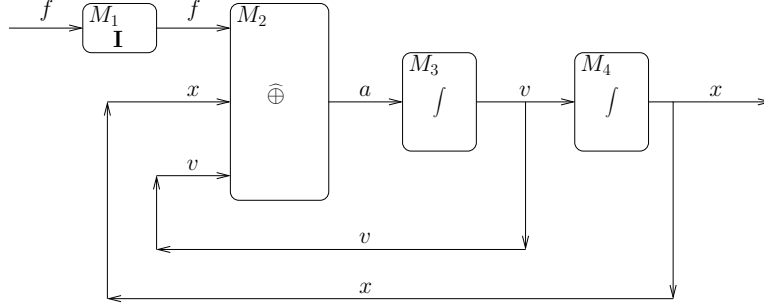


Figure 3. Case study: the network

There are 3 other modules M_2, M_3, M_4 , with associated functions F_i ($i = 2, 3, 4$):

$$\begin{aligned} a(t) &= F_2(f, x, v)(t) = (f(t) - Kx(t) - Dv(t))/M \\ v(t) &= F_3(a)(t) = \left(\int_0^t a\right) + v_0 \\ x(t) &= F_4(v)(t) = \left(\int_0^t v\right) + x_0 \end{aligned}$$

The integration constants v_0 and x_0 represent initial velocity and displacement.

4.4 Network semantics. The parameter list is $\mathbf{c} = (M, K, D, v_0, x_0)$, the single *input stream* is f , and the list of *non-input streams* is $\mathbf{u} = (a, v, x)$. So we want a fixed point of the function $\Psi_{\mathbf{c}, f} : \mathcal{C}[\mathbf{T}, \mathbb{R}]^3 \rightarrow \mathcal{C}[\mathbf{T}, \mathbb{R}]^3$, where $\Psi_{\mathbf{c}, f}(a, v, x) = (a', v', x')$ with

$$\begin{aligned} a'(t) &= (f(t) - Kx(t) - Dv(t))/M \\ v'(t) &= \left(\int_0^t a\right) + v_0 \\ x'(t) &= \left(\int_0^t v\right) + x_0. \end{aligned} \tag{4.1}$$

For changes $\delta a, \delta v, \delta x$ in a, v, x , and corresponding changes $\delta a', \dots$ in a', \dots :

$$\Psi_{\mathbf{c}, f}(a + \delta a, v + \delta v, x + \delta x) = (a' + \delta a', v' + \delta v', x' + \delta x').$$

Then (using the pseudonorm $\|u\| =_{df} \sup \{u(t) \mid T \leq t \leq T + \tau\}$) from (4.1):

$$\|\delta a'\| \leq (K\|\delta x\| + D\|\delta v\|)/M \tag{4.2a}$$

$$\|\delta v'\| \leq \tau\|\delta a\| \tag{4.2b}$$

$$\|\delta x'\| \leq \tau\|\delta v\|. \tag{4.2c}$$

Now assume $M > \max(K, 2D)$ (4.3)

and put $\kappa =_{df} \max(K, 2D)/M$, (4.4)

$\tau =_{df} D/M$. (4.5)

By (4.3), $\kappa < 1$. Define the product pseudonorm $\|(\delta a, \delta v, \delta x)\| =_{df} \|\delta a\| + \|\delta v\| + \|\delta x\|$. (This corresponds to taking $p = 1$ in (3.1).) Then

$$\begin{aligned} \|(\delta a', \delta v', \delta x')\| &= \|\delta a'\| + \|\delta v'\| + \|\delta x'\| \\ &\leq (K/M)\|\delta x\| + (D/M + \tau)\|\delta v\| + \tau\|\delta a\| \quad \text{by (4.2)} \\ &\leq \kappa \|(\delta a, \delta v, \delta x)\| \quad \text{by (4.4) \& (4.5)} \end{aligned}$$

which proves **Contr**_{c,f}(κ). Note that the only assumption needed to prove the contraction property was (4.3), i.e., that the mass M be sufficiently large relative to the stiffness K and damping coefficient D . No assumption was needed on either the initial values v_0 and x_0 of velocity and displacement, or the external force $f(t)$. Hence, from Theorem 2:

Theorem 3. *The network of Figure 3 is contracting, and hence satisfies (NetDet), for any input stream $f(t)$, provided $M > \max(K, 2D)$.*

Corollary. *The system of Figure 2 has a well-determined solution $(a(t), v(t), x(t))$ for the acceleration, velocity and displacement as functions of time $t \geq 0$, for any input force $f(t)$ as a continuous function of time $t \geq 0$, and any initial conditions (v_0, x_0) for the velocity and displacement, provided only that $M > \max(K, 2D)$. Moreover, under this condition, the solution streams (a, v, x) depend continuously on the input stream f and the parameters (M, K, D, v_0, x_0) .*

5 Computability of the solution

We want to show that the network function which solves the network specification (2.2) according to Theorem 2 is computable relative to the module functions; in other words, the output streams are computable from the input streams, parameters, and module functions. Hence if the module functions are computable, then so is the network function.

By “computable” here we mean: computable according to some concrete model of computation on $\mathcal{C}[\mathbb{T}, A]$. We give a new model, inspired by the approximation of $\mathcal{C}[\mathbb{T}, A]$ by the metric spaces $\mathcal{C}[[0, k], A]$.

An alternative treatment of concrete computation on the space $\mathcal{C}[X, Y]$ with the compact-open topology is given in [Wei00], with $X \subseteq \mathbb{R}^m$ and $Y = \mathbb{R}^n$.

5.1 Topological algebra of streams. Consider the *5-sorted topological algebra*

$$\mathcal{C} = (A, \mathbb{R}, \mathbb{T}, \mathcal{C}[\mathbb{T}, A], \mathbb{N}; \mathbf{d}, \text{eval})$$

where $\mathbf{d} : A^2 \rightarrow \mathbb{R}$ and $\text{eval} : \mathcal{C}[\mathbb{T}, A] \times \mathbb{T} \rightarrow A$ are, respectively, the distance function on A and the evaluation function: $\text{eval}(u, t) = u(t)$. \mathcal{C} is a topological algebra, because each of the five carriers has an associated topology with

respect to which the basic functions (\mathbf{d} and \mathbf{eval}) are continuous. The carrier \mathbb{R} is needed for the metric on A . The set of sorts of the signature Σ of \mathcal{C} is $\mathbf{Sort} = \mathbf{Sort}(\Sigma) = \{A, R, T, C, N\}$. For ease of notation, we also refer to the five carriers of \mathcal{C} as C_s for $s \in \mathbf{Sort}$.

5.2 Enumerations of subfamilies of \mathcal{C} . The following extends the concepts in [TZ04] on concrete computation on metric algebras to the case of the topological (non-metric) algebra $\mathcal{C}[\mathbf{T}, A]$. We will fix an *enumeration* of certain subsets of the carriers, i.e., a family α of bijections $\alpha_s : \mathbb{N} \rightarrow X_s \subseteq C_s$ ($s \in \mathbf{Sort}$) of \mathbb{N} with certain subsets X_s of C_s . The pair (X_s, α_s) is called an *enumerated subset* of C_s . The enumerations are as follows.

First, the mapping $\alpha_A : \mathbb{N} \rightarrow X \subseteq A$ is an enumeration of some dense subset X of A . Here we need a separability assumption:

(Sep): A is separable.

From **(Sep)** follows that $\mathcal{C}[\mathbf{T}, A]$ is also separable. This enumeration α_A (or rather its “computational closure” $\bar{\alpha}_A$, see below) must also satisfy a Σ -*effectivity* assumption, to be described below (5.4.3). The mapping $\alpha_{\mathbb{R}} : \mathbb{N} \rightarrow \mathbb{Q} \subset \mathbb{R}$ is a standard enumeration of the rationals. (In case $A = \mathbb{R}$, α_A is the same as $\alpha_{\mathbb{R}}$.) Similarly $\alpha_{\mathbb{T}} : \mathbb{N} \rightarrow \mathbb{Q}^+ \subset \mathbb{T}$ is a standard enumeration of the non-negative rationals. The mapping $\alpha_{\mathbb{N}}$ is just the identity on \mathbb{N} . Finally, and most interestingly, the mapping $\alpha_C : \mathbb{N} \rightarrow Z \subset \mathcal{C}[\mathbf{T}, A]$ is a “standard” enumeration of some countable dense subset Z of $\mathcal{C}[\mathbf{T}, A]$, which must satisfy a Σ -*effectivity* assumption (5.4.3 below), as well as the following:

Assumption 5.2.1 (Effective locally uniform continuity of (Z, α_C)).

There is a recursive function $\mu : \mathbb{N}^3 \rightarrow \mathbb{N}$ such that for all k, ℓ, n , writing $z_n = \alpha_C(n)$:

$$\forall t_1, t_2 \in [0, k] : |t_1 - t_2| < 2^{-\mu(k, \ell, n)} \Rightarrow \mathbf{d}(z_n(t_1), z_n(t_2)) < 2^{-\ell}.$$

5.3 Computational closure. For our model of *concrete computation* on $\mathcal{C}[\mathbf{T}, A]$, we construct the *computational closures* $\mathcal{C}_{\alpha_s}(X_s)$ of the enumerated subsets (X_s, α_s) of the spaces C_s , with enumerations $\bar{\alpha}_s : \Omega_{\bar{\alpha}_s} \rightarrow \mathcal{C}_{\alpha_s}(X_s)$, so that $X_s \subseteq \mathcal{C}_{\alpha_s}(X_s) \subseteq C_s$ for $s \in \mathbf{Sort}$, as we now describe.

First, for the metric space A , we define the set $\mathcal{C}_{\alpha_A}(X)$ of α -*computable elements* of A , to be the limits in A of effectively convergent Cauchy sequences of elements of the enumerated subset X , with corresponding enumeration $\bar{\alpha}_A$. Details of the construction of $\mathcal{C}_{\alpha_A}(X)$ and $\bar{\alpha}_A$ can be found in [TZ04]. We omit them, since below, for the computational closure of $Z \in \mathcal{C}[\mathbf{T}, A]$, we describe a model of concrete computability for a more general situation — the non-metric topological space $\mathcal{C}[\mathbf{T}, A]$.

The computational closures $\mathcal{C}_{\alpha_{\mathbb{R}}}(\mathbb{Q})$ and $\mathcal{C}_{\alpha_{\mathbb{T}}}(\mathbb{Q}^+)$ in \mathbb{R} and \mathbb{T} respectively are defined in the same way. The computational closure of \mathbb{N} is, trivially, \mathbb{N} ,

with (again) the identity enumeration. Finally, for the space $\mathcal{C}[\mathbb{T}, A]$ with its enumerated subset (Z, α_C) (where we henceforth usually drop the subscripts of α and $\bar{\alpha}$), let $\mathcal{C}_\alpha(Z) \subset \mathcal{C}[\mathbb{T}, A]$ be the set of all limits in $\mathcal{C}[\mathbb{T}, A]$ of α -effectively locally uniform Cauchy sequences of elements of Z — such limits always existing by the completeness of $\mathcal{C}[\mathbb{T}, A]$ (Lemma 3.1.2) — and let $\Omega_{\bar{\alpha}} \subset \mathbb{N}$ be the set of codes for $\mathcal{C}_\alpha(Z)$. More precisely, $\Omega_{\bar{\alpha}}$ consists of pairs of numbers $c = \langle e, m \rangle$ where (i) e is an index for a recursive function defining a sequence

$$z_0, z_1, z_2, \dots \quad (5.1)$$

of elements of Z , where $z_n = \alpha(\{e\}(n))$; and (ii) m is an index for a modulus of local uniform convergence, i.e., $\forall k, \forall n, p \geq \{m\}(k) : \mathbf{d}_k(z_n \upharpoonright_k, z_p \upharpoonright_k) \leq 2^{-k}$. For any such code c , $\bar{\alpha}(c)$ is defined as the limit in $\mathcal{C}[\mathbb{T}, A]$ of the Cauchy sequence (5.1), and $\mathcal{C}_\alpha(Z)$ is the range of $\bar{\alpha}$.

5.4 Concrete computation on $\mathcal{C}[\mathbb{T}, A]$. For a tuple of sorts $\sigma = (s_1, \dots, s_m)$ we have the product space $C^\sigma =_{df} C_{s_1} \times \dots \times C_{s_m}$, the product domain $\Omega_{\bar{\alpha}}^\sigma =_{df} \Omega_{\bar{\alpha}_{s_1}} \times \dots \times \Omega_{\bar{\alpha}_{s_m}} \subseteq \mathbb{N}^m$, and the product enumeration $\bar{\alpha}^\sigma = (\alpha_{s_1}, \dots, \alpha_{s_m}) : \Omega_{\bar{\alpha}}^\sigma \rightarrow C^\sigma$.

Definition 5.4.1 (Tracking function). Let $f : C^\sigma \rightarrow C_s$. A function $\varphi : \Omega_{\bar{\alpha}}^\sigma \rightarrow \Omega_{\bar{\alpha}_s}$ is an $\bar{\alpha}$ -tracking function for f if the following diagram commutes:

$$\begin{array}{ccc} C^\sigma & \xrightarrow{f} & C_s \\ \bar{\alpha}^\sigma \uparrow & & \uparrow \bar{\alpha}_s \\ \Omega_{\bar{\alpha}}^\sigma & \xrightarrow{\varphi} & \Omega_{\bar{\alpha}_s} \end{array}$$

Definition 5.4.2 (Concrete computability on $\mathcal{C}[\mathbb{T}, A]$). Suppose f, g_1, \dots, g_k are functions on $\mathcal{C}[\mathbb{T}, A]$ with $\bar{\alpha}$ -tracking functions $\varphi, \psi_1, \dots, \psi_k$ respectively. Then f is $\bar{\alpha}$ -computable in (or relative to) g_1, \dots, g_k iff φ is partially recursive in ψ_1, \dots, ψ_k .

We need one more assumption on the enumeration α .

Assumption 5.4.3 (Σ -effectivity of $\bar{\alpha}$). The basic functions of the algebra $\mathcal{C}[\mathbb{T}, A]$, namely \mathbf{d} and \mathbf{eval} , are $\bar{\alpha}$ -computable.

Example 5.4.4 (Concrete computation on $\mathcal{C}[\mathbb{T}, \mathbb{R}]$). Consider, in particular, the case that the metric space A is \mathbb{R} . As stated above, for α_A we would take the same as $\alpha_{\mathbb{R}}$, i.e., a standard enumeration of the rationals.

As an example of a countable dense subset of $\mathcal{C}[\mathbb{T}, \mathbb{R}]$, take $Z = \mathbb{Z}\mathbb{Z}$, the set of all continuous “zigzag functions” from \mathbb{T} to \mathbb{R} with finitely many turning points, all with rational coordinates. Clearly, the set $\mathbb{Z}\mathbb{Z}$, under any reasonable enumeration α_C , satisfies the effective locally uniform continuity assumption (5.2.1). Also, the enumeration $\bar{\alpha}$ derived from α is clearly Σ -effective (Assumption 5.4.3).

We could use instead, as our starting point, the set of polynomial functions of t with rational coefficients. This yields the same set $\mathcal{C}_\alpha(Z)$ of computable elements of $\mathcal{C}[\mathbb{T}, \mathbb{R}]$.

5.5 Relative concrete computability of functions defined by analog networks. Given a network N , we want to show the network function Φ^N is $\bar{\alpha}$ -computable relative to the module functions, provided it is contracting at the parameter and stream inputs.

For this we need a constructive concept of contraction, namely that a contracting factor $\kappa < 1$ can be found *effectively* in the parameters and stream inputs \mathbf{c}, \mathbf{x} over some domain.

Definition 5.5.1 (Effectively contracting network). Given $U \subseteq \mathcal{C}[\mathbb{T}, A]^p \times A^r$, the network N is $(\bar{\alpha})$ -effectively contracting on U if a contracting factor $\kappa_{\mathbf{x}, \mathbf{c}}$ can be found $\bar{\alpha}$ -effectively in $(\mathbf{x}, \mathbf{c}) \in U$.

Note that this certainly holds with the case study in Section 4, where a value for κ can be found effectively in the parameters M, K, D (and independent of the input stream f), by equations (4.3) and (4.4), in the region $U =_{df} \{(M, K, D) \in \mathbb{R}^3 \mid M > \max(K, 2D)\}$.

Theorem 4. Suppose the network N satisfies **(Caus)**, (Z, α_C) satisfies effective local uniform continuity, and $\bar{\alpha}$ is Σ -effective. Suppose also N is $\bar{\alpha}$ -effectively contracting on $U \subseteq \mathcal{C}[\mathbb{T}, A]^p \times A^r$. Then the network function Φ^N is defined (at least) on U , and is $\bar{\alpha}$ -computable relative to the module functions of N . Hence if the module functions are $\bar{\alpha}$ -computable, then so is Φ^N .

Proof (outline). For an input $(\mathbf{x}, \mathbf{c}) \in U$, the output of Φ^N is a sub-tuple of the fixed point \mathbf{u} of $\Psi_{\mathbf{c}, \mathbf{x}}^N$ (§3.2). So it suffices to show that the function from $(\mathbf{x}, \mathbf{c}) \in U$ to this \mathbf{u} is computable. (Here “computable” means $\bar{\alpha}$ -computable relative to the module functions.)

Consider the sequence of stream tuples \mathbf{u}_n , defined in the proof of Theorem 1, with $f = \Psi_{\mathbf{c}, \mathbf{x}}^N$. First, each \mathbf{u}_n is computable in (\mathbf{x}, \mathbf{c}) , by induction on n .

Further, (\mathbf{u}_n) is an *effectively* locally uniform Cauchy sequence, i.e. (in the notation of Definition 3.1.1) N can be obtained effectively from k . From this it follows that the limit \mathbf{u} of this sequence, which is the fixed point of $\Psi_{\mathbf{c}, \mathbf{x}}^N$, is also computable in (\mathbf{x}, \mathbf{c}) . \square

5.6 Concrete computability of module functions. The standard module functions on $\mathcal{C}[\mathbb{T}, \mathbb{R}]$ are $\bar{\alpha}$ -computable. For (a) *pointwise addition* and (b) *scalar multiplication* this is obvious. The interesting case is (c) *integration*. Here, in taking the integral as the limit of a Cauchy sequence of Riemann sums, we use the *effective locally uniform continuity* assumption (5.2.1).

Thus all the module functions in the case study in Section 4 are concretely computable. Combining this with Theorem 4, we conclude that the function which solves the network equations in that example is concretely computable.

6 Concluding Remarks

Most current research on analogue systems is focused on computation on the reals with the traditional processing units (adders, integrators etc.). Our network models, involving arbitrary processing units on data from metric spaces in continuous time, are new.

Several questions and problems are left open:

1. For the modules, it turned out that we did not need the assumption of *time invariance* (satisfied by the standard module functions in §5.6) which, like *causality* (which we did need) is common in dynamical system theory [OW97]. What is the significance of this assumption — or its absence?
2. What if we allow *partial* or *many-valued* module functions or partial streams?
3. Find reasonable conditions, other than the contraction property, that guarantee “good behaviour” of these networks.
4. Characterise the networks that produce all (and only) computable functions on $\mathcal{C}[\mathbf{T}, A]$.

References

- [Bus31] V. Bush. The differential analyzer: a new machine for solving differential equations. *Journal of the Franklin Institute*, **212** (1931) 447–488.
- [Eng89] R. Engelking. *General Topology*. Heldermann Verlag, 1989.
- [GC03] D.S. Graça and J.F. Costa. Analog computers and recursive functions over the reals. *Journal of Complexity*, **19** (2003) 644–664.
- [Gri99] E. Griffor, editor. *Handbook of Computability Theory*. North Holland, 1999.
- [Hyn70] D.E. Hyndman. *Analog and Hybrid Computing*. Pergamon Press, 1970.
- [Kle52] S.C. Kleene. *Introduction to Metamathematics*. North Holland, 1952.
- [MC04] J. Mycka and J.F. Costa. Real recursive functions and their hierarchy. *Journal of Complexity* **20** (2004) 835–857.
- [Moo96] C. Moore. Recursion theory on the reals and continuous time computation. *Theoretical Computer Science* **162** (1996) 23–44.
- [OW97] A.V. Oppenheim and A.S. Willsky. *Signals and Systems (2nd edition)* Prentice Hall, 1997.
- [PE74] M.B. Pour-El. Abstract computability and its relation to the general-purpose analog computer. *J. American Mathematical Society* **199** (1974) 1–28.
- [Sha41] C. Shannon. Mathematical theory of the differential analyser. *Journal of Mathematics and Physics* **20** (1941) 337–354.
- [TT80] W. Thompson and P.G. Tait. *Treatise on Natural Philosophy (2nd edition) Part I*. Cambridge University Press, 1880.
- [TT91] B.C. Thompson and J.V. Tucker. *Algebraic specification of synchronous concurrent algorithms and architectures*. Research Report CSR 9-91, Department of Computer Science, University College of Swansea, 1991.
- [TZ94] J.V. Tucker and J.I. Zucker. Computable functions on stream algebras. In: *Proof and Computation: NATO Advanced Study Institute Summer School at Marktoberdorf, 1993*, ed. H. Schwichtenberg. Springer-Verlag (1994) 341–382.
- [TZ04] J.V. Tucker and J.I. Zucker. Abstract versus concrete computation on metric partial algebras. *ACM Trans. on Computational Logic*, **5** (2004) 611–668.
- [Wei00] K. Weihrauch. *Computable Analysis: An Introduction*. Springer-Verlag, 2000.