JET PROPULSION LABORATORY

INTEROFFICE MEMORANDUM

331.1-95-045
November 20, 1995

To: Distribution

From: Dave Watola

Subject: Overview of the Downlink Analyzer

REFERENCES:

1. J. B. Hampshire II, *A Differential Theory of Learning for Efficient Statistical Pattern Recognition*. PhD. thesis, Carnegie Mellon University, Department of Electrical & Computer Engineering, September 1993.

2. D. Watola, "Overview of the Downlink Analyzer Front End," JPL IOM 331.1-95-046, November 20, 1995.

3. D. Watola, "Overview of Downlink Analyzer Neural Network Technology," JPL IOM 331.1-95-047, November 20, 1995.

4. D. Watola, "Differential Discriminative Diagnosis for Downlink Analyzer," JPL IOM 331.1-95-048, November 20, 1995.

5. D. Watola, "Some Open Issues in Downlink Analyzer," JPL IOM 331.1-95-049, November 20, 1995.

1. Purpose

This document is the first in a series of five memoranda detailing the background theory, implementation, and issues surrounding the Downlink Analyzer task. This memo is an introduction to Downlink Analyzer, and is intended to provide the reader with a brief, high-level view of its objectives and components. It also serves as a reference point to anchor the remaining four papers in the series [2-5], to which explicit details are deferred.

2. Downlink Analyzer Overview

Downlink Analyzer is a tool for performing model-based fault diagnosis using a non-parametric neural-network model. Currently, it is targeted to fault identification applications in Deep Space Network downlink subsystems, but its principles are applicable to a broad class of problems.

A high-level block diagram of Downlink Analyzer operation is shown in Figure 1. Only the topmost level of abstraction and processing are considered in this section. Individual lower-level blocks are described in subsequent sections.

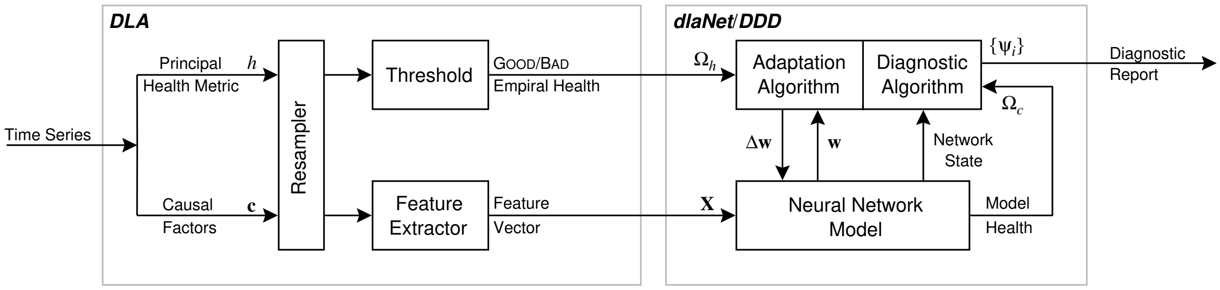

Figure 1. Downlink Analyzer block diagram.

Raw time series data are accepted by Downlink Analyzer at the left of Figure 1. Typically, the input data comprise several streams of monitor data from a Monitor & Control subsystem. Only one restriction is placed on the content of this data. For each system subject to fault diagnosis, the data must be partitionable into a single *principal health metric*, h, and a set of *causal factors*, $\{c_i\}$. There are no other constraints on the nature or content of the data, which need not even be synchronously or uniformly sampled.

A principal health metric should be selected such that a simple thresholding function, $\Omega_h(h)$, can be formulated to yield the empirical health state of the system, Ω_h. Downlink Analyzer adopts the convention that $\Omega_h = \Omega_1$ denotes the GOOD or *healthy* state while $\Omega_h = \Omega_0$ indicates a BAD or *anomalous* condition. Generally, the empirically-derived Ω_h is taken to be equal to the actual system health state Ω even though h is a stochastic process subject to underlying random process noise, measurement noise, and time delays. Thus the true mapping from h to Ω is characterized by a set of posterior probabilities $\{P(\Omega_i | h)\}$, and Ω_h is merely an estimate of Ω. For a principal health metric that is frequently updated and estimated with high statistical confidence, the simplification proves viable.

The elements of $\{c_i\}$ are selected such that an arbitrarily complex function $\Omega_c(\mathbf{c})$ can be constructed that also produces the correct value $\Omega_c = \Omega$ without examining h. The purpose of Downlink Analyzer, then, is to dissect this more complicated model $\Omega_c(\mathbf{c})$ and determine the cause of a transition from healthy state Ω_1 to anomalous state Ω_0 even when the causal relationships in question are obscure or heretofore unknown. Results of this *post mortem* are a set of *saliencies* $\{\Psi_i\}$ providing a quantitative measure of the contribution from each c_i to the degraded performance. The final ranked list of causal factors constitutes the output of the complete Downlink Analyzer system. Meaningless analyses can be avoided in regions where the model is not robust since the true health state (or at least a robust estimate of the most likely actual health state) is always available in Ω_h for comparison.

An underlying assumption of the diagnostic procedure is that components of $\mathbf{c}$ possess enough diversity to engender an accurate model of the target system. Omission of a salient causal factor, whether due to oversight or lack of availability, precludes effective fault diagnosis when that factor is a principal cause, but is otherwise harmless since model inadequacy is readily apparent from any discrepancy between Ω_h and Ω_c. The inclusion of redundant or irrelevant factors can degrade model robustness in the presence of novel inputs, but procedures for pruning unnecessary complexity from the model exist to ameliorate this phenomenon.

Finally, it is also important that $\{c_i\}$ data be sufficiently low-level measurements to permit useful diagnoses. For example, the $\{c_i\}$ subset {azimuth pointing error, elevation pointing error, system noise temperature (SNT), received carrier power (P_c)} is preferable to the singleton set $\{P_c/N_0\}$ since the aim of diagnosis is isolation of the lowest-level cause(s) of an anomaly; in this case, P_c/N_0 is better suited to the station of principal health metric.

The conceptual steps required to produce saliency rankings from raw data are discussed in the following three sections.

3. Front End Processing (*DLA*)

The large gray box at the left of Figure 1 represents the portion of Downlink Analyzer responsible for pre-processing input data before presentation to the model and for producing the empirical health state estimate, Ω_h. In the prototype implementation, the contents of this block are embodied in a software tool called ***DLA***.

Once the raw data is split into a principal health metric and all other causal factors, each data stream passes through a user-specified resampling process. The resampler can leave the data unmodified, or optionally can resample the streams on an individual basis. Using this mechanism, time series can be synchronized with each other or to an independent source of reference events (e.g. every five seconds). Provisions exist for inserting (repeating) samples when the underlying process does not emit them often enough, and for recognizing the start of a large gap. In ***DLA***, the most common application of resampling forces all data streams to be synchronously (but not necessarily uniformly) sampled.

After resampling, the principal health metric is transformed into an empirical health state by comparing it to a user-supplied threshold. Threshold selection is usually based on empirical observations and human experience with the target system.

Resampled raw data do not themselves comprise the input to the model. Instead, the front end provides a mechanism for computing features based on resampled data. Features are user-defined, arbitrarily complex functions of the instantaneous (last-known) values of the resampled data. More importantly, features can also be derived from resampled data statistics measured over a feature-specific, user-selected temporal window.

Feature vectors, shown as **X** in Figure 1, and empirical health state estimates are always issued simultaneously. Timing of these emissions is independent of resampling, but is controlled in essentially the same fashion. Typically, outputs appear either at regular intervals or synchronous with the arrival of new raw samples of the principal health metric.

For detailed information about resampling algorithms, feature extraction, and ***DLA*** consult [2].

4. Learning Model (*dlaNet*)

The mapping from $\mathbf{X}$ to Ω_c is implemented by a neural network as shown in the rightmost gray box in Figure 1. Application of a connectionist model sidesteps some of the challenges associated with identifying a suitable parametric model for the system and determining an appropriate model order from empirical data. Instead, it provides a general-purpose non-parametric framework for modelling the desired process. Of course, literally non-parametric models are an illusion–the connection weights $\mathbf{w}$ in the neural network model clearly comprise a set of parameters. However, the neural network paradigm represents a non-parametric model in the same sense that classical (i.e. periodogram-based) spectral analysis techniques yield non-parametric frequency-domain process models: the parameters are coefficients of basis functions spanning a particular hypothesis class.

Connection weights in the network are determined by an adaptation algorithm that is essentially an optimization procedure. In Downlink Analyzer, the ***dlaNet*** tool is used to train the network with a *training set* of feature vector/empirical health state pairs carefully selected to be representative of the underlying input space. Ideally, the training set provides uniform coverage of the multidimensional input space such that if the network successfully learns those patterns, it will exhibit reasonably good generalization performance when confronted with novel feature vectors during its normal operating regime. Note that ***dlaNet*** does *not* include the diagnostic algorithm block in Figure 1. Instead, ***dlaNet*** processing occurs offline, orthogonal to any real-time operational use.

In addition to driving the weights toward an acceptable model based on training data, ***dlaNet*** also incorporates techniques for network regularization. Regularization removes unnecessary weights, i.e. those that do not contribute significantly to correct classification performance, from the network in order to reduce complexity and improve generalization.

Complete details concerning the neural network, adaptation algorithm, and regularization techniques used by Downlink Analyzer are found in [3]. Further useful background information is located in John Hampshire's thesis [1].

5. Diagnostic Model (*DDD*)

During normal operation, the weights of the neural network model are fixed so the adaptation algorithm block in Figure 1 is not needed. Instead, the diagnostic algorithm block is present. In the Downlink Analyzer prototype, the neural network core and the diagnostic procedure are contained in a tool called ***DDD***.

An overview of the diagnostic procedure has already been presented above; for an in-depth consideration of the theory behind the algorithm, see [4].

Distribution:

M. Belongie	238-420
F. Pollara	238-420
S. Townes	238-420
N. Xiao	238-420
D. Watola	82-104