A note on the Gamma test.

by

Aðalbjörn Stefánsson[1], N. Končar and Antonia J. Jones

Abstract.

This note describes a simple technique, the Gamma (or Near Neighbour) test, which in many cases can be used to considerably simplify the design process of constructing a smooth data model such as a neural network. The Gamma test is a data analysis routine, that (in an optimal implementation) runs in time O(*M*log*M*) as *M* -> ∞, where *M* is the number of sample data points, and which aims to estimate the best Mean Squared Error (MSError) that can be achieved by any continuous or smooth (bounded first partial derivatives) data model constructed using the data.

Keywords: Dynamic systems, Prediction, Control, Feedforward Neural Networks, Delta-test.

Communication regarding this paper should be addressed to:

Antonia J. Jones

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF WALES, CARDIFF
PO BOX 916
Cardiff CF24 3XF

Telephone: ..+44-292-087-4812
Telefax: ..+44-292-087-4598

Published in *: Neural Computing & Applications* (1997), **5**:131-133.

[1] This name is of Icelandic extraction and so the first name must be given in full.

A note on the Gamma test

CONTENTS

A note on the Gamma test

by

Aðalbjörn Stefánsson[1], N. Končar[2] and Antonia J. Jones[3]

Abstract. This note describes a simple technique, the Gamma (or Near Neighbour) test, which in many cases can be used to considerably simplify the design process of constructing a smooth data model such as a neural network. The Gamma test is a data analysis routine, that (in an optimal implementation) runs in time O($M\log M$) as $M \to \infty$, where M is the number of sample data points, and which aims to estimate the best Mean Squared Error (MSError) that can be achieved by any continuous or smooth (bounded first partial derivatives) data model constructed using the data.

Keywords: Dynamic systems, Prediction, Control, Feedforward Neural Networks, Delta-test.

Introduction.

In this short note we describe a simple algorithm which we have found exceedingly useful in our work. We do not describe a mathematical justification for the algorithm, since it precisely constructing such a mathematical statement and proof which has caused the main paper describing the algorithm and giving example applications to be considerably delayed. However, it is clear from the analysis that a suitable theorem and proof can indeed be provided and the debate has mainly centered about the most general hypotheses which ensure that the conclusions remain valid. Because we believe that the implications of this technique are far ranging we decided to make the basic idea and software generally available to a wider audience.

The Gamma test

Let a data sample be represented by

$$((x_1,\ ...,\ x_M),\ y)\ =\ (\boldsymbol{x},\ y) \tag{1}$$

in which we think of the vector $\mathbf{x} = (x_1, ..., x_m)$ as the *input*, confined to a closed bounded set $C \subseteq \mathbb{R}^m$, and the scalar y as the *output*. In the interests of simplicity the following explanation is presented for a single scalar output y. But the same algorithm can be applied to the situation where y is a vector with very little extra complication or time penalty. The Gamma test is designed to give a data-derived estimate for Var(r).

We focus on the case where samples are generated by a suitably smooth function (bounded first and second order partial derivatives) f: $C \subseteq \mathbb{R}^m \to \mathbb{R}$ and

$$y\ =\ f(x_1,\ ...,\ x_m)\ +\ r \tag{2}$$

where r represents an indeterminable part, which may be due to real noise or might be due to lack of functional determination in the posited input/output relationship i.e. an element of `one -> many-ness' present in the data.

We make the following assumption

[1] Department of Physics, Chalmers University of Technology, Gothenburg, Sweden.

[2] Department of Computing, Imperial College of Science, Technology and Medicine, 180 Queen's Gate, London, SW7 2BZ, UK.

[3] Department of Computer Science, University of Wales - Cardiff, PO Box 916, Cardiff CF2 4YN, UK.

- **Assumption A**. We assume that training and testing data are different sample sets in which: (a) the training set inputs are non-sparse in input-space; (b) each output is determined from the inputs by a deterministic process which is the same for both training and test sets; (c) each output is subjected to statistical noise with finite variance whose distribution may be different for different outputs but which is the same in both training and test sets for corresponding outputs.

Suppose ($\mathbf{x}$, y) is a data sample. Let ($\mathbf{x}'$, y') be a data sample such that $|\mathbf{x}' - \mathbf{x}| > 0$ is minimal. Here $|.|$ denotes Euclidean distance and the minimum is taken over the set of all sample points different from ($\mathbf{x}$, y). Thus $\mathbf{x}'$ is the nearest neighbour to $\mathbf{x}$ (in any ambiguous case we just pick one of the several equidistant points arbitrarily). The Gamma test (or near neighbour technique) is based on the statistic

$$\gamma = \frac{1}{2M}\sum_{i=1}^{M}(y'(i) - y(i))^2 = \frac{1}{2}\langle(y'(i) - y(i))^2\rangle \qquad (3)$$

It can be shown that γ -> Var(r) in probability as the nearest neighbour distances approach zero. In a finite data set we cannot have nearest neighbour distances arbitrarily small so the Gamma test is designed to estimate this limit by means of a linear correlation.

Given data samples ($\mathbf{x}(i)$, $y(i)$), where $\mathbf{x}(i) = (x_1(i), ..., x_m(i))$, $1 \leq i \leq M$, let $\mathbf{x}(N(i, p))$ be the p th nearest neighbour to $\mathbf{x}(i)$. Nearest neighbour lists for p*th* nearest neighbours ($1 \leq p \leq p_{max}$), typically we take p_{max} in the range 20-50, can be found in O(Mlog$M$) time using techniques developed by Bentley, see for example [Friedman 1977].

We write

$$\Delta(p) = \frac{1}{p}\sum_{h=1}^{p}\frac{1}{M}\sum_{i=1}^{M}|\mathbf{x}(N(i, h)) - \mathbf{x}(i)|^2 \qquad (4)$$

and

$$\Gamma(p) = \frac{1}{p}\sum_{h=1}^{p}\frac{1}{2M}\sum_{i=1}^{M}(y(N(i, h)) - y(i))^2 \qquad (5)$$

then Δ(p) is the mean square distance of the $h \leq p$ nearest neighbours and Γ(p) is an estimate for the statistic γ (defined in (3)) based on the $h \leq p$ nearest neighbours.

The Gamma Test algorithm is then

```
Procedure Gamma (or Near Neighbour Test) (data)

      (* data is an array of points (x(i), y(i)), (1 ≤ i ≤ M), in which
      x is a real vector of dimension m and y is a real scalar *)

For i = 1 to M (* compute x-nearest neighbour list for each data point
*)
      For p = 1 to pmax
            N(i, p) = t where x(t) is the p th nearest neighbour to
            x(i).
      endfor p
endfor i

For p = 1 to pmax
      compute Δ(p) as in (4)
      compute Γ(p) as in (5)
endfor p

Perform least squares fit on coordinates (Δ(p), Γ(p)) (1 ≤ p ≤ pmax)
```

```
obtaining (say) y = Ax + Γ̄
Return (Γ̄, A)
```

Discussion.

The theoretical debate concerns the precise circumstances under which the correlation between $\Delta(p)$ and $\Gamma(p)$ can be proved to be approximately linear when $\Delta(p)$ is small. This depends on the stochastic sampling process of the input data vectors **x** (the sampling must be such as to ensure that nearest neighbour distances become small as M -> ∞), on the distribution of *r* having reasonable properties (such as bounded variance) and on the smoothness properties of the function *f*. Still, the fact remains that we have yet to find any non-pathological examples satisfying these conditions where, given sufficiently many data points, the correlation fails to be linear. In practical applications one should in any event always examine the regression line, which sometimes reveals unexpected structure in the data.

A technique which allows one to estimate Var(*r*), on the hypothesis of an underlying continuous or smooth model *f*, is of considerable practical utility in applications such as control or time series modelling. The implication of being able to estimate Var(*r*) in neural network modelling is that then one does not need to train the network (or indeed *any* smooth data model) in order to predict the best possible performance with reasonable accuracy.

We have used the Gamma test to:

- To find the minimal number of data samples required to produce a near optimal model. We do this by computing Γ for increasing *M*. The graph asymptotes to the true value of Var(*r*). When M is sufficiently large to ensure that Γ has stabilised close to the asymptote there is little advantage to be gained by increasing *M*.

- To automatically and rapidly (the near neighbour information can be used to speed up training by picking the data points with worst errors and performing backpropagation on a suitable subset of the weights) construct a near minimal neural network architecture and weights which best models the data [Končar 1997]. The Gamma test provides the criterion for ceasing training.

- To determine the best embedding dimension and delay time for time series [Masayuki 1997].

- To determine the best set of inputs from a list of possible inputs for a neuro-controller [Končar 1997].

The last two applications illustrate what is perhaps the main utility of the Gamma-test: on data sets which are not excessively large the test is sufficiently fast to be run on a compete examination of all possible subsets of up to 20 inputs (for a larger number of inputs we use a Genetic Algorithm for which the fitness of a selection of inputs is based on how small the Γ value is for the selection).

The Gamma test software has been implemented in *C* using long double precision arithmetic to allow accurate processing of very large high precision data sets. It can be downloaded with instuctions from http://www.cs.cf.ac.uk/ec

References

[Friedman 1977] J.H. Friedman, J.L. Bentley and R.A. Finkel. *An algorithm for finding best matches in logarithmic expected time.* ACM Transactions on Mathematical Software, **3**(3):200-226, 1977.

[Končar 1997] N. Končar. *Optimisation methodologies for direct inverse neurocontrol*. Forthcoming Ph.D. thesis, Department of Computing, 180 Queen's Gate, London, SW7 2BZ, U.K.

[Otani 1997] Masayuki Otani. *Guiding chaotic orbits*. Research Report, Department of Computer Science, University of Wales, Cardiff, PO Box 916, Cardiff CF2 3XF, Wales, U.K.