Subject: Help: Coding iterative algorithm elimimating loops in IDL/PV~WAVE
Posted by black on Wed, 09 Sep 1992 13:35:04 GMT
View Forum Message <> Reply to Message

Has anyone got an idea how to implment the following algorithm in IDL/PV~WAVE that DOESN'T use for loops. The reason for the avoidance of for loops is because, as everyone should know, for loops are slow.

The algorithm is a training algorithm for vector quantisers, which can be viewed as a sort of unsupervised neural network; one application of vector quantisers is data compression.

There are two sorts of vectors. Training vectors and code vectors. Training vectors are chosen from the population of vectors that the code vectors are to represent. The number of code vectors is finite and very much less than the number of vectors in the population that the training vectors are a subset of. Data compression (which is lossy due to the finite number of code vectors) is achieved by the storing or transmitting of an integer which represent a code vector. Reconstruction consists of looking up the code vector which corresponds to a given integer.

The aim of the training algorithm is to minimise the the squared error of representing the training vectors, given a set of code vectors. The code vector which is chosen to represent a given training vector is the code vector that is closest to the training vector.

The training algorithm goes like this

1) Pick a training vector at random from the set of training vectors

2) Find the code vector that represents the training vectors

3) Change the value of the code vector so it moves towards the training vector by amount delta (delta is between zero to one)

4) Repeat steps 1 to 3 until the total error is acceptably small, or doesn't change significantly

This algorithm is known as a continuous training algorithm, and performs gradient descent.

The reason why the repetition of steps 1 to 3 CAN'T be done in parallel is that that the changing of the position of a code vector in one run through steps 1 to 3 might change which code vectors represent a training vector on the next run through the steps. Thus the algorithm is intrinsically iterative, which suggests for loops. So I suppose the real question is either

1) Given that array-type operations are faster than equivalent sequential

operations, is there a way to implement a series of iterations using array operations which reproduce the series of iterations

2) Is there a fast way to implement iterations in IDL/PV~WAVE, ie one that avoids do loops?

There is an equivalent batch algorithm that re-estimates the code vectors by for all code vectors, centroiding all the training vectors that are represented by a given code vector. This algorithm can be implemented in IDL/PV~WAVE using array type operations.

My current implementation of batch algorithm in PV~WAVE is 12 to 30 times faster than continuous algorithm in PV~WAVE, depending on the value of delta. The reason why I'd like to speed up the continuous training algorithm is so that I can compare the two algorithms properly.

Anyone any sugestions, other than write the continuous training algorithm in 'C'. Please E-mail responses direct to me.

Thanks in advance,

John Black.

Janet: jvb@uk.mod.hermes
Internet: jvb%hermes.mod.uk::relay.mod.uk