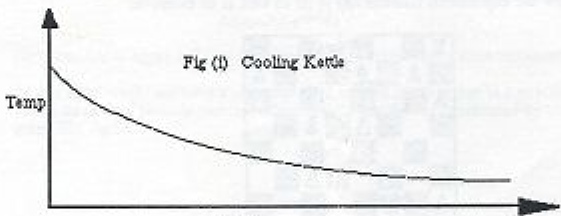
Evaluating Chess Computer Programmes
A New Approach

Over the years there have been many successful systems devised to quantify the quality of chess played by an individual. The most popular is the method devised by Professor Arpad E Elo. This uses statistical principals to evaluate the relative strength of a chess player based upon their past results. The ELO method is used by FIDE.as well as by Eric to obtain accurate ratings for players and computers alike. However the main drawback is the amount of time required to obtain an accurate result. It may surprise you to know that there are on average about 4500 man hours of work required to evaluate the playing strength of a computer to within 10 ELO points!! Now in human competition this level of accuracy is never really required since a persons actual playing strength will vary from day to day and tournament to tournament. However in the case of a computer the quality of its chess is fixed and quite often there can be up to ten different machines all within 10 ELO point range. Moreover when a new machine comes onto the market there is a great temptation for the manufacturers to claim slightly optimistic playing strengths, knowing that it will be quite a long time before their claims can be refuted.

There have been several attempts to design a method of evaluation aimed at cutting down the time required to estimate the playing strength of a computer. The schemes seem to fall into one of two categories. The first is '*find the move*' where the computer is asked to find a move which is definitely winning, usually either a mating sequence or material gain. However the main drawback to this is that games are not usually won or lost on just one move. There needs to be consideration given to positional build-up, pressure and timing before the clinching combination can be found and executed. Any evaluation system which ignores these factors is going to loose accuracy as a result. The second type of system is of the '*How Good is Your Chess*' type. Here the computer is asked to go move by move through a game, trying to find the next move played by a Grandmaster. This means that the computer is asked to consider many different positions from early middle game to end-game. In may cases this system can work very well. However there are still drawbacks in so much as the computer's style might clash with that of the Grandmaster's and as a result the computer may obtain an undeservedly low score.

There is also a need for some form of evaluating system which distinguishes between playing strengths at different speeds of play; something not even the ELO system can easily do. The system I am about to describe solves many of our problems. It uses an *exponentially decaying reward* to give a score to how well a computer copes with analysing a position. Now if you have not come across exponential functions before they are nothing to shy away from; we encounter them every day even though we do not realise it. For example when a kettle of boiling water cools down to room temperature it follows an exponential decay path (see fig i).

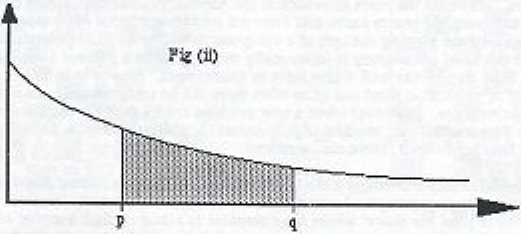


Other processes which are governed by the exponential function are radioactive decay, certain electronic components and many other naturally occurring systems. The exponential function, let us call it $f(x)$, is defined mathematically as;

$$f(x)=A.e^{-dt} \qquad \text{Equation (1)}$$

where A and d are constants and e is the exponential constant (2.71828). As A and d vary, the shape of the curve changes.

The exponential function has several interesting properties, one of which is the area under the curve between two points.



This area can be calculated exactly by using the following formula (the derivation of this is given in the appendix for those who are interested);

$$Area=\frac{A}{d}(e^{-dp}-e^{-dq}) \qquad \text{Equation (2)}$$

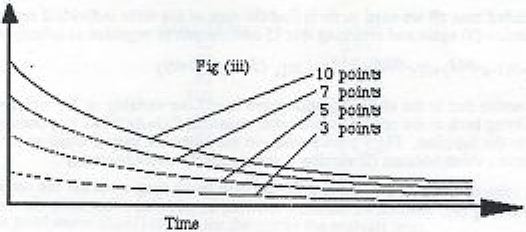
where A and d are the constants and p,q are the bounds of the area which you want to find (see fig ii). Note also that by using this formula we find the area under the graph between zero and infinity is given by A/d (since $e^{-0}=1$ and $e^{-infinity}=0$).

You may well be asking what does all of this have to do with evaluating chess programmes? The link is the *'Beat The Masters'* column published in *Pergamon Chess*. Every month a team of chess professionals examines six to ten ordinary chess positions (ie not necessarily containing tactical combinations) from actual games. They first decide individually which move they consider the best and why. Then they all collectively decide on which move is actually the strongest. This move is awarded ten points and all other reasonable moves obtain a score less than ten, depending upon their relative value. The results are then published two months after the original positions were set. To see how our exponential function ties in let us look at an example.

White to play

The results of the above position were published in *Pergamon Chess* December 1988 as position 'J' and were as follows; *Rae1 10; Nxf5 7; Nce2, f4, Bf4, Bg1 5; Rg1 3; Others 0.* Two computers, Saitek's *Stratos* and Martyn Bryant's *Colossus IV* were given the position to analyse for 15 minutes.

Stratos chose *Nxf5* from start to finish. However *Colossus IV* was not so certain, playing *Nxf5* for the first 1 minute 34 seconds, then *a4* until 2 minutes 35 seconds and finally *Rae1* for the rest of the test. It would seem fair to give *Stratos* 7 points; since no matter what level it was set to, it would have played a move of value 7 points. However what score do you attribute to *Colossus IV*? A possible solution can be found by using exponential functions as a weight. Imagine we have four curves, such that the areas below the curves between zero and infinity correspond to the move score 10, 7, 5, and 3 respectively. We will call this area from zero to infinity, for any particular curve, the *curve's value*. Fig (iii) shows a representation of four such curves. Assume that the horizontal axis represents time.



Now this is the critical part to understand - *for each move considered find the curve whose value is that of the move. For the time period that it is considered calculate the area under the this curve.* This means that for *Colossus IV* we would first of all want to find the area under the curve of value 7 (the value of *Nxf5*) between the points zero and ninety four (1 minute 34 seconds converted to seconds). If this is repeated for each move considered by the computer in its 15 minute analysis then the sum of these areas can be used as a score for the computer. This is the fundamental idea behind the system.

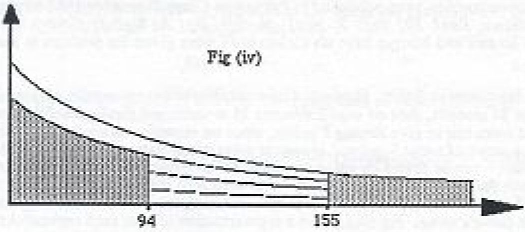
As an example let us work through to find *Colossus IV's* score for the above position. Using equation (2) we can work out the relevant areas exactly. So for *Colossus IV's Nxf5* move we would have;

$$Area=7(e^{-0}-e^{-94d})$$

$$Area=7(1-e^{-94d})$$

Note that *A/d* is equal to 7 (see equation 3) and the times have been converted into seconds.

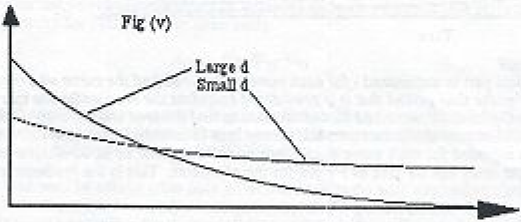
At this stage we do not have a value for *d* but we will come to that in a moment. We can repeat this for all three time periods corresponding to the three moves considered by *Colossus IV*, to obtain a graph like fig iv.



To find the total shaded area all we need to do is find the sum of the three individual sections of area. Applying equation (3) again and assuming that 15 minutes can be regarded as infinity we get;

$$Area=7(1-e^{-94d})+0(e^{-94d}-e^{-155d})+10(e^{-155d}-e^{-infinity})$$

However you will notice that in the above equation there is still the variable *d*, for which we do not have a value. Going back to the original exponential equation, $f(x)=Ae^{-dt}$,we can changed and see what happens to the function. Fig v shows what we get when we vary *d* while at the same time keeping the curve's value constant (ie varying *A* and *d* such that *A/d=constant*)



We can see that the effect of changing *d* is to change the functions *emphasis*. This means that if *d* is large then there is more emphasis placed upon the early part of the function, while if *d* is small the emphasis moves towards the latter part of the function. Therefore if we could find some suitable method of correlating *d* to the rate of play, then we would could use equation (3) to give a score which incorporates the rate of play.

Suppose we have a rate of play of *R* seconds per move. We can alter the value of *d* such that a certain proportion of the total area (remember total area = curve value = area from zero to infinity) falls under the function between zero *R* seconds. This proportion, let us call it *P*, and the rate of play *R* are related to the variable *d* by equation (4). Again the proof is in the appendix.

$$d=\frac{1}{R}\log_e(1/P) \qquad \text{Equation (4)}$$

Now by experimenting with different values for *P*, Eric and myself have found that *P=(1/2.25)* seems to place the correct emphasis upon the various parts of the function. So now, if we know the rate of play we can use equation (3) to give a score to *Colossus IV* for the example we used earlier. For a rate of play equal to one move every ten seconds (ie $R=10$) we can find d;

$$d=\frac{1}{10}\log_e(2.25)$$

$$d=0.08109$$

We can now substitute this value into equation (3) to give us a score for *Colossus IV* at 10 seconds per move.

$$Score=6.997$$

This would seem very reasonable since if *Colossus IV* encounters this position in ten second chess it would play a move worth 7 points. If we change R, d and hence the score will correspondingly change.

Rate Of Play	Score
10	6.997
30	6.600
60	6.266
120	6.800
180	7.391
240	7.828
300	8.148

As you can see the score goes down then up. This is because the mathematics are accounting for the relatively poor move (*a4*) played between 1 minute 34 seconds and 2 minutes 35 seconds and then the good move (*Rae1*) chosen for the rest of the analysis time.

If the computers were to analyse about twenty five to thirty of such positions then an average percentage score could be given for each machine at different rates of play. This would lead to a range of scores associated with different rates of play. One slightly less obvious advantage of this is that it could help to direct the programmers in development. If there is a large range of scores then this indicates that if the programme was speeded up (increased processor power) then the playing strength will also increase significantly. However if the range of scores is small it would probably be more worthwhile for the programmers to concentrate upon increasing the computer's chess knowledge.

The accuracy of this system has yet to be established and will no doubt depend much upon the 'quality' and number of positions used. One problem will be that not all the positions will be of equal difficulty for the computers to analyse. Therefore it is likely that a weighting factor will have to be given to each position in order to reflect its difficulty.

The weaknesses of the system would seem to lie in three areas. Firstly it does not account for efficient use of time by the computer. Some programmes have particularly good time management systems. These can allow the computer to save time while analysing relatively straight-forward positions, enabling the computer to search for longer in more complex situations where more time can be beneficially used. Secondly , the size and quality of the opening book can improve a programmes performance. These factors are not accounted for when evaluating the computers since it is only the raw 'move search' part of the programme which is analysed. The last weakness is the complexity of the system. Unlike other methods of evaluation, this system is quite complex to fully understand how it works ('A' level maths standard) and it requires a reasonable amount of concentration to use it. However it is not essential to understand the system fully, only an ability to apply the equations and even this can be programmed into a computer.

On page 8 there are four positions which you can try out on your chess computers; we suggest that you allow twenty minutes for each position to be analysed. In the next issue of the *News Sheet* there will be another twenty four such positions which you will be able to test your computers with and then send us (Eric & myself) the results to analyse. The results will then be published in a future edition of the *News Sheet*..

The system is still being developed and I would certainly appreciate it if there were comments and suggestions from readers as to improvements, possible positions etc

Steve Maughan

Friday 10 November 1989

Appendix

The Exponential Function

An exponential function is one of the form;

$$f(x)=Ae^{-dt}$$

Where d is the timing constant and A is a constant of proportionality. Integral calculus allows us to calculate the area under the function.

$$Area=\int_a^b f(t).dt$$

Where a and b are the two points which define the bounds of the area on the t axis.

$$Area=\int_a^b Ae^{-dt}.dt$$

$$Area=\left[\frac{Ae^{-dt}}{-d}\right]_a^b$$

$$Area=A(e^{-da}-e^{-db})/d \qquad .Equation\ (1)$$

To find the area under the function from zero to infinity, let a=0 and b=infinity;

$$Area\ from\ zero\ to\ infinity=A/d$$

Assume that R is the rate of play, expressed in seconds per move. To find a relationship between R, A and d, we must equate the area between zero and R with 1/2.25 (found empirically). This equation can then be solved for R.

From equation (1) $\quad Area=A(e^{-da}-e^{-db})/d$

Therefore $\quad A/(d.2.25)= Area=A(e^{-d0}-e^{-dR})/d$

Rearrange $\quad 1/2.25=1-e^{-Rd}$

Therefore $\quad e^{-Rd}=1/2.25$

Take Logs (base e) $\quad -Rd=Log_e(1/2.25)$

$$d=Log_e(2.25)/R$$

$$\underline{d=0.8109/R}$$

The above techniques can be found in any good 'A' level maths text book under the headings of exponential functions, integration and logarithms.

Four Positions For You To Try!

White to play

Marks: ♘a3 *10;* ♘d4 *7;* ♕e2 *5;* ♘xc7, ♘xd6 *3;* a4 *2;* Others *0.*

White to play

Marks: e5 *10;* ♕g4, ♕c2 *5;* ♕f3, ♗h3, b4 *3;* Others *0.*

White to play

Marks: ♖c7 *10;* ♖xc8, ♘d4 *6;* f5 *3;* h3, ♗b2, ♘c3, ♘g3, ♕d4 *2;* Others *0.*

Black to play

Marks: ♗a4 *10;* ♗g4 *9;* ♗e6 *8;* ♗a3 *5;* g6, ♗xd4 *1;* Others *0.*