Iterative method for static analysis of a Go position: A few ideas

Daniel Hu 6d

02 Dec 2020

Abstract

Influence functions have been considered for a long time in Go to make an estimate of which player controls which positions on a Go board, and hence aid in judgement of a position. From professionals counting territory to GnuGo estimating control to amateurs drawing fancy pictures to the modern precise results of Katago's control estimate. In this article I investigate some possible heuristics derived from considerations of symmetry to make an estimate of control based on an iterative algorithm, analogous to numerical solutions of differential equations.

The algorithm is very simple with no learning, pattern matching, or reading and hence is very inaccurate with no understanding of life and death and yet is quite slow. But the heuristics might be interesting, the outputs are nice to look at and there is some improvement on simple propagating functions: evaluating stones captured in a net as dead, judgement over the reach of influence, closer to solutions of endgame corridors.

1 Introduction

How do we judge a Go game?

Go is made up of only a few basic components: the $19 \times 19 = 361$ intersections on the board, the graph that represents connections between them, the legal moves, and more abstractly, the game tree of all possible legal variations. And finally each leaf node of the game tree, i.e. when the game ends, is given a score. Even if this score isn't necessarily algorithmic, being based on a difficult judgement of which stones are alive and dead, computers can use the Tromp-Taylor rules in which all dead stones must be *proved* dead by being taken off the board in order to count towards the capturer's score. Other than these final moves at the end of the game, this is close to equivalent to standard human rulesets for the purposes of defining optimal play and the score obtained by it. In this way, we can discuss the *score under optimal play* of a board position that is still far from being finished. Neural networks feed the final result into training positions encountered in self-play games in order to make the bots stronger at recognising which positions lead to better results. In this way the information at leaf nodes is propagated back through the game tree and absorbed as a function of patterns that arise in positions long before the endgame.

It has been popular to further granularise score into predicting the control of each intersection on the board. Especially in more "pure" rulesets such as Chinese rules, the total score is determined by the difference in the total control that the two players have on the board[1]. The total control is a simple sum of the intersections you have stones on or have surrounded at the end of the game[2]. One of the reasons is that in Go there is such a thing as unconditional life, a point where a group is certain under optimal play (at least locally) to be able to remain alive forever[3]. At this point, we can pretty much already conclude which side will control those intersections and any territory they surround as eyespace. And hence, regarding everything that happens in between putting a stone on the board and making it unconditionally alive, we can consider it in terms of fractions as progress towards making it alive, though the details are too complicated to describe with a single number.

In go, as long as a group isn't unconditionally alive, there is always the possibility that it will become convenient to sacrifice it when the opponent attacks, generally to avoid dying with even more stones. Therefore, it seems incorrect to assume that a stone is alive as soon as you place it on the board just because you want to commit to it. You might not have the opportunity to commit to it, so it seems correct to think in terms of probabilities when it comes stones that aren't yet part of an unconditionally alive group (which requires a set of a minimum of 6 stones working together).

[1] "On the board" means before taking komi into account.

[2] The Japanese ruleset is almost equivalent, but more "practical" with rules for special situations.

[3] There is an even stronger notion of pass-alive in which your stones are alive no matter what you opponent does, even if you keep passing and not responding to attacks.

Hence we are interested in control estimate functions that tell us for each intersection on the board the likelihood it will be controlled by white or black. We can then use this for evaluating the current board position and score and also compare different possible future board positions, as well as working out where the interesting areas to play on are. In this article, I will use the standard convention of 1 to represent full black control of an intersection and -1 to represent white control. Numbers in between represent the intermediate probabilities scaled accordingly.

1.1 GnuGo influence function

GnuGo isn't a particularly strong bot anymore. It doesn't use reinforcement learning and its main source of strength is pattern matching. It mainly bases its move evaluation on an influence function. This takes each living stone on the board and draws a function around it that reduces exponentially with the distance to the source stone. This function is computed by propagation one step at a time and therefore can take into account the flow of influence being blocked by stones of the opponent.

It has the notions of permeability and attenuation that govern the extent to which influence is propagated from one intersection to a neighbour.

Permeability is a notion associated with when a move is sente so that the influence is blocked off because the opponent will defend in response to a move that attacks a group or area. In this case permeability is zero and the influence stops propagating through. The example of the space in a one space jump is given, because the owner of it is very likely to respond to a peep and the opponent is unlikely to cut. Hence despite there being a space for the opponent's influence to flow through, a scenario where the opponent cleanly cuts through is quite unlikely, reducing the degree to which influence on one side affects the play on the other side.

Attenuation is the factor by which influence reduces with each step. Arguably this is a more general form of permeability, but when the opponent is thick nearby, you normally have to play more solidly to stay connected, so your influence reaches less far, but if you are thick nearby, you have further reach and attenuation may be lower.

1.2 Professional territory estimate

Consider for example Cho Chikun's book Positional Judgement High-speed Analysis that has many diagrams marking out the territory that each side has claimed. The boundaries are normally drawn assuming you respond to every reducing move the opponent plays from the outside. One potential problem is that it doesn't think too much about more aggressive moves by the opponent. Also, the endgame can be flawed if you don't work out which colour is more likely to reduce an area at the boundary in sente, and you only assume the most submissive variations for each side. Of course, the major flaw is the inability to count influence in areas which haven't yet solidified into territory. This was always very difficult before the AI.

1.3 Kataestimate

Katago's neural network gives a score estimate for each position it is fed, and also an estimate for how likely each of the 361 intersections is to be controlled (occupied or as part of territory) by black or white at the end of the game. Then, by averaging over large numbers of playouts, it can display its best estimate of score and control for the current board position based on its analysis from a high depth of reading. This is an amazing tool that the creator says has vastly increased the efficiency of its training[2]. It is a great visualisation tool, not available in its competitors to the best of my knowledge at the time of writing, and its main advantage is its accuracy and precision, being in combination with a very strong bot.

If there is a weakness, it is that although consolidated territories will normally still be held by a player on all the main lines, the estimate for regions which are being fought over can be very biased to Katago's own style of play and how it would play against itself. Often, once the number of playouts surpasses say 1 million, an overwhelming majority of its playouts in any position are focused just on one or two moves. These aren't necessarily the same as what Alphago might play (from reviewing Alphago self-play games), and certainly not the same as what a human would play. The estimate presented will be the average of the estimates on the lines that it looks into.

2 Static analysis

Arguably it is somewhat behind the times to investigate relatively "dumb" methods that use neither lots of pattern matching like GnuGo, or reinforcement learning such as AI. There is no way to match the accuracy of heavy training and deep reading. However, I think there may still be plenty of value to be found for methods based on endgame theory. One possible advantage of such a method is more "understanding" and insight rather than working with a relatively black box.

In this line, I was inspired by Bill Spight's lecture on thermography which discusses the endgame theory of corridors and how the influence of a reduction stone reduces by a factor of two in such cases, and then discusses the possibilities and difficulties of applying this to influence functions in more general positions[4].

The goal of *static analysis* is to construct a function or algorithm to predict the position by position control of the board given a fixed board position. The restriction given by "static" is that no *reading* should be done, and all analysis should be done based on that position, or loosening such a condition a little, just looking at most n moves ahead where $n \sim 5$ is small.

This is surely a tough task as ot my knowledge there is not even a good algorithm to solve for the score of a final board position except under computer rulesets such as Tromp-Taylor where all dead stones must be taken off the board in order to make them count towards your score. This is because working out the life and death of chains and whether an area is territory or not under optimal play is probably an NP-complete problem? (find source)

However, we can still try to find heuristics that work in certain situations perhaps by inspiration from principles from endgame theory. For example, if an endgame move can be played by either player, then we generally estimate the score by assuming there is a 50-50 chance it will be taken by either player. In particular, the expected score from *corridors* that are open at only one end can be calculated precisely in this manner, as well as many standard endgame situations. The probability that the invader intrudes n steps inside is proportional to 2^{-n}.

We can express this principle in terms of influence functions, by imagining the influence of an invading stone reducing by a factor of 2 for every step it enters the opponent's area. In a corridor, you can only invade one step at a time or await your opponent cutting your move off and rendering it useless. So how do we extend such ideas to endgame moves where the follow up isn't adjacent to the initial move but further in, perhaps with the support of nearby stones of the same colour. We could call this "reach". And how do we understand how different supporting stones interact to generate possible endgame moves. For example a corridor open at two ends isn't counted simply as twice the endgame from two ends, but instead slightly less than double as the two ends are partially redundant. The two influence functions flow past each other, so the estimate for each position isn't a simple function of the neighbouring estimates but also must take into account the directions of the flow. In addition when they overlap, they make the other redundant. This is analysed in more detail below.

One major difficulty for static analysis is sente moves, which standard endgame theory handles in a somewhat crude manner. If a follow up move is larger than the initial move played in an area, we say it is sente, and normally this means that if it is optimal for the first player to make the initial move there, then it is near optimal play for the second player to respond immediately. This could be because the first player's follow up threatens to save or capture a larger group of stones or invade or block off a larger area of territory. Unfortunately, it normally requires a lot of reading to work out which moves are sente.

Another difficulty is measuring the life and death of stones even approximately, and counting that in the calculations. After all dead stones will have much less influence on the game (but not necessarily zero if they short the opponent's liberties).

I have written some programs to perform static analysis. Diagrams are drawn in grayscale.

2.1 Neural networks

We can take some inspiration from neural networks and it would be very interesting though tedious to investigate their inner workings. They pack a huge amount of data and "knowledge" into their network weights, so pulling apart the "organs and tissues" of their workings would probably be instructive.

Convolutional neural networks consider the 19x19 board as a table of numbers and iteratively applies rounds of *filters*. The parameters for the filters can be tuned in the training process. For each of the intersections, these collate the numbers at neighbouring cells (i.e. intersections), take the dot product of these with some vector (which is fixed for that round, and called the filter) and place that new number at that intersection. There are some edge effects that are normally smoothed out by zeroing out the entries that come from beyond the edge of the board.

The nice property that this filter is the same for all the intersections adds simplicity, meaning less parameters, and also significantly speeds up computation time with parallel processing methods and GPUs. In Go, this also meaningfully represents the translational symmetry of the rules that don't depend on which intersection you are at, ignoring some edge effects. On the other hand, you shouldn't read too much into this as Deepmind have successfully applied the same techniques to games without such symmetry such as Chess and Shogi[3].

Neural networks have a form of parallel processing built into the architecture, so that the output of one filter is fed along many different routes as the input to many other filters, so the information propagates down many different "neural pathways" in the network. From the perspective of each board of numbers, as many rounds of filters are applied, information about the board position propagates from the local to the global one step at a time. The first few filters might highlight local board positions that are only dependent on the placement of stones immediately adjacent to each intersection, while later filters will combine this information to give a global judgement of the position.

In a way, a neural network does compute a static analysis for each position, without any direct reading because the information from reading ahead is encoded into the parameters in the training process. For each position in a self-play game, the training updates the parameters based on the result of that game. For kataestimate, it will learn the information about which colour controlled which intersections in the final position. A single neural network evaluation is a static function. But only when combined with lots of reading in a Monte Carlo Tree Search does AI become overwhelmingly strong.

The problem is that we don't understand what the neural networks are really doing, as they are very complicated, even if we can examine certain neurons and describe what sort of patterns they are activated by. Part of the problem is that there are many different equivalent ways for a neural network to compute the same function, so that even if you can describe what sort of output you think the network is producing, and understand what sort of components are required for that computation, the network might fuzzily combine those components in a complicated way. This lack of a one-to-one correspondence is quite confusing for humans to deal with.

But we can at least draw hope from the fact that there is a function based simply on repeated calculations that are the same across the whole board that eventually outputs a very accurate answer.

2.2 Propagating function

A propagating influence function is designed to mirror the way the rules about liberties propagate across the board. The only way a difference in position on one side of the board can influence proper play on the other side of the board (other than in a combinatorial game theory way[4]) is by the "influence" it exerts across the board. A stone affects the liberties of stones placed next to it whether your own or the opponent's which then affects the strength and weakness of stones placed next to those stones and so on. In particular we have the following principle.

> "Influence cannot travel through strong groups"

The meaning of this is that neighbouring stone makes no difference to strength (i.e. potential to be captured) of a strong group. Hence in particular, if a strong group cuts the board into two or more regions, then what happens on one side of the strong group is independent of what happens on the other side.

GNUGo uses a propagating function, so I coded up something similar. In Figure 2, the analysis image is shown for the game in Figure 1. Each stone on the board is assumed to be alive (because working out life and death is difficult), and influence propagates from it in each of the 4 compass directions, attenuating by a factor of 2 each time and not passing through stones on the board, of either colour. Influence is capped between -1 (for a white stone) and 1 (for a black stone) and presented in grayscale proportionally. It may actually be a better idea to cap the total black influence and white influence separately before summing them, see the next subsection on corridors, but for the image I merely capped the total value.

The many captured white stones (white is behind this game) are not understood by this analysis, though at least their influence won't propagate far, being surrounded. It also doesn't understand the weakness of white's positions

[4]Say we define dependent games as those where one move might affect both games simultaneously, and independent games are ones where the moves are completely separate. The point is that only way the two sides of the board can be dependent is if they "influence" each other and are connected by the rules somehow.

But even two independent games can still affect each other by changing the optimal play when playing both simultaneously. This is due to the possibility of sacrificing something in one game (i.e. one side of the board) in order to gain in the other game (the other side of the board). However, in our case, we are merely looking for an averaged estimate of control rather than an exact estimate. We are interested in miai counting rather than optimal play. This greatly mitigates the impact of complicated CGT possibilities.

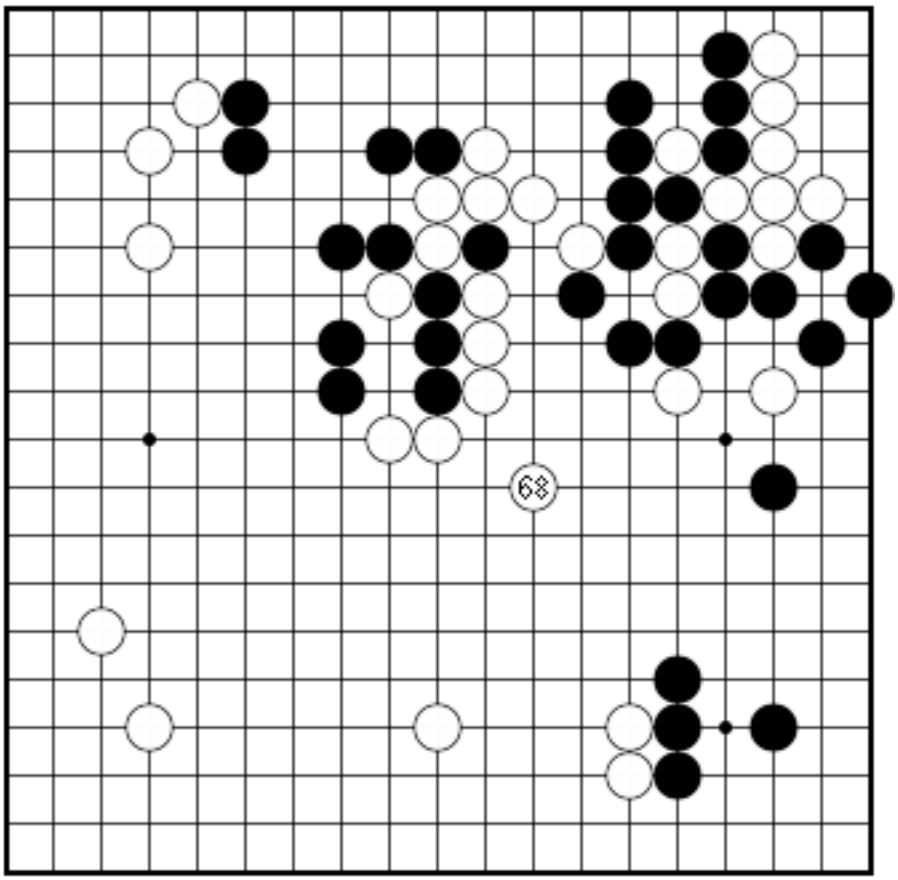
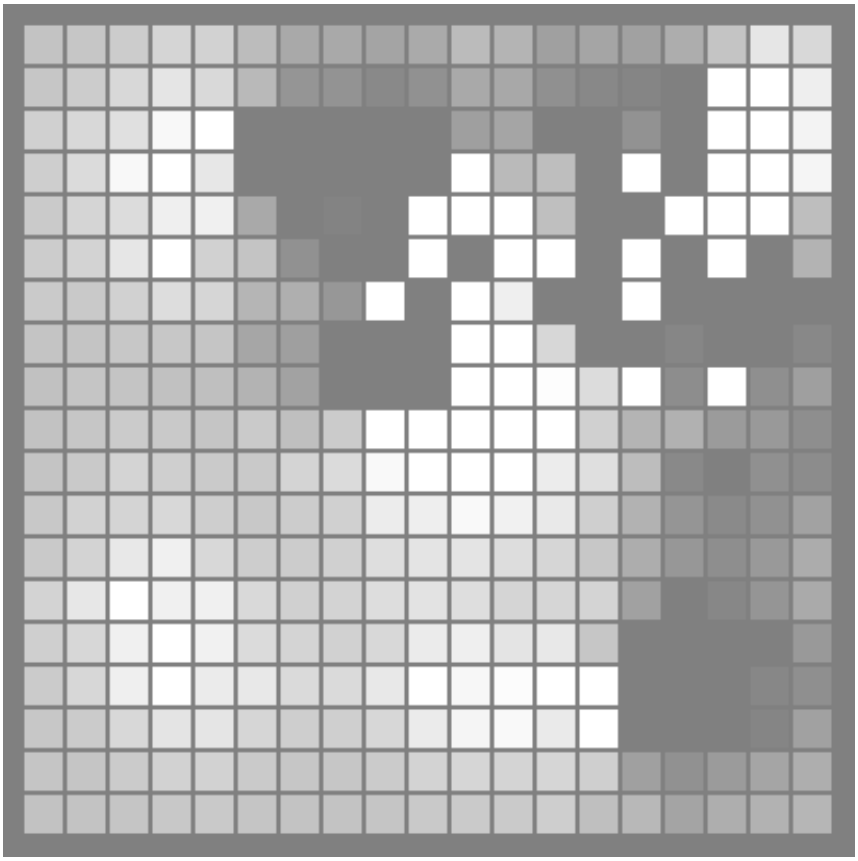


Figure 1: Chinese City A League 2020 Round 1 - Hu YaoYu 8p vs Xie Ke 8p



W+12.75

Figure 2: Propagating function analysis

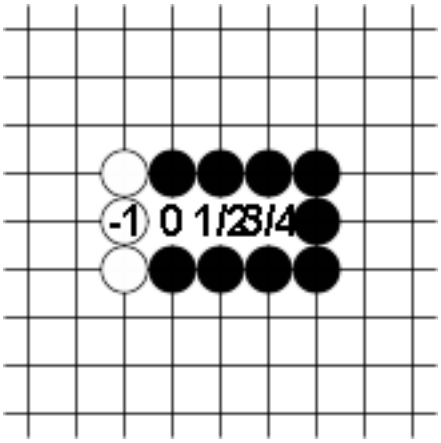


Figure 3: Corridor, Influence functions

in the upper left, upper right and lower middle, being so close to black's thickness. Although the estimates feel like they have some justification behind them as each side is more likely to control the areas closer to their stones, the influence functions don't seem to surround areas of territory. For example the areas in the corners are mostly surrounded and fairly likely to be controlled the player dominating them at the moment, but the influence rapidly fades to a 50-50 score of zero by the time it reaches the 1-1 corner point. This is in part because this fixed function doesn't consider sente moves. In the process of surrounding territory, players will focus on keeping the corner and will be more likely to respond to the opponent invading rather than leaving it as 50-50.

Problems that arise:

1. The influence easily overflows past -1 or 1, and just capping it at these values is rather crude.
2. How to judge whether stones are alive?
3. How should the influence functions of difference stones of either colour interact? Is it really just a simple sum?
4. How to deal with sente sequences where stones are expected to be placed in certain positions given certain patterns in the board position. GnuGo makes a partial patch by adding virtual stones and their influence functions at standard invasion points (by pattern matching).

2.3 Endgame principles

2.3.1 Corridor

A corridor in Go endgame theory is a series of empty intersections connected in a row (possibly with turns) placed between two walls completely controlled by one colour. Normally the two walls are one space apart, called a width one corridor. Normally the two ends are blocked off by stones of opposing colours. This is called a corridor open at one end. See Figure 3. We assume all stones currently on the board are unconditionally alive. In this shape, due to the lack of space, white has no hope of living inside the corridor unless white connects (solidly) to the stone on the left marked with a "-1" (due to it being a completely alive white stone). In the same vein, black just has to add one move in the corridor to prevent white from entering any further, at which point black claims the rest of the corridor enclosed by that stone as territory. In the initial position black can make two points of territory by playing at the point marked by "0".

Recall that we are using 1 to denote full black control, -1 full white control and 0 a 50-50 position.

The way we count this using (the more accurate) miai counting is to average the position between what happens if black plays next and if white plays next. So there are equal chances for the players to play at the point marked by "0". If black claims it then there is no more useful endgame left locally. If white claims it, then their next follow up is to push in one step further at the point marked "1/2" which happens with half of the previous chance so 1/4 in total. Hence our calculation gives $1/4(-1)+3/4(1) = 1/2$ as expected. The final move at the point marked "3/4"

is dame, but is still worth fighting for in area counting, and in any case has an even chance of being controlled by either player if white has already pushed in twice.

"Reducing moves in your opponent's area have half the influence for every successive move"

We have this principle, but it isn't clear how far "each move" reaches. In a corridor you have to stay solid and move to a neighbouring point one step at a time. But in more empty positions, it is more normal to reduce by one space jumps and knight's moves. In this way, you can enter two each time (and so the reduction effect of halving influence extends for more space).

Note that if we consider propagating functions in a way where we sum the influence function for black capped to 1 and sum the influence function for white capped to -1, doing the capping before the final sum of black and white influence, then they will give the correct form of solutions for the corridor. This is because the influence function that the invader contributes halves with each step. The defender will always contribute the maximal amount as it is in their area.

2.4 Generalising questions

As well as the questions above, we must also ask

1. How does influence propagate on a more open board?
2. How does the influence of different stones interact? In particular how does it interact with weak stones?

In answer to the second, we can make the point that the extent to which a stone's influence flows through a weak group is based on the difference it makes on the strength of the group. This impacts on how threatening follow up moves on the other side of the group are. There will be no impact past a completely dead group or a completely alive group. A move will make a lot of difference in practice if it stops an enemy group from making eyes or supports the eyespace of your own group.

2.5 More corridors

Consider Figure 4. In general, for a length n corridor open at one end, the invader's influence penetrates a total of $\sum_{k=1}^{n-1} 2^{-k} = 1 - 2^{-(n-1)}$ points (territory counting). This is relative to the score if black blocks off the open end, taking $n-1$ points. For a length n corridor open at both ends, the invader's influence penetrates a total of $1+\sum_{k=1}^{n-3} 2^{-k} = 2-2^{-(n-3)}$ points, relative to if black blocks off both ends, taking $n-2$ points. This is slightly less than double that for when the corridor was open at one end by a margin of $2^{-(n-3)}$. The difference is mostly due to overlapping influence but also partially due to the miai situation between white and black's first move. White's reducing stones are just slightly in conflict because there is the possibility that one of them will be wasted if black doesn't respond and lets white enter from both sides under each reduces an area which already has no points. This occurs after white spends $n-3$ moves in the corridor more than black. The reduction factor is lessened the further apart the two white stones are from each other, with less overlap of their influence.

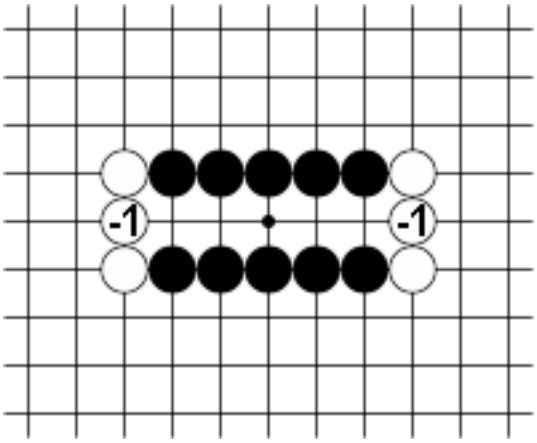
This demonstrates that it is hard to surround territory. There are diminishing returns to having multiple threats to reduce the same area of the opponent.

"Each extra move that reduces the same area of the opponent is worth less than the previous one."

It is most natural to apply this to the centre which is open on all sides to being reduced. It gets less and less valuable to have access to your opponent's centre area if you only need one strong group to reduce it. This might in part explain why AI likes the 3-3 invasions so much. It is confident it still has just enough space and aji to reduce the resulting walls.

2.6 Sente

In endgame theory, a move is considered sente if the follow up is larger (in miai value) than the initial move. In this case the simplest way to analyse the position is to assume that the person threatening the follow up will always make the initial move, and the other player will always respond. Given this, the rest of the game tree is analysed in the usual way. Moves are called miai if the initial move and the follow up have the same value, so that whichever



The first two moves are miai, which is equivalent to sente for miai counting calculations. Hence play in a length n corridor open at both ends reduces to play in a length $n-2$ corridor open at one end. Without loss of generality, we can assume play starts on the left. If white pushes from the left. Then the corresponding control estimates are $(1, 0, \frac{1}{8}, \frac{1}{4}, \frac{1}{2})$ in terms of probability they are controlled by white. If black defends on the left, we obtain $(0, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1)$. Averaging these including with the reflective symmetry we obtain $(\frac{5}{8}, \frac{7}{32}, \frac{3}{16}, \frac{7}{32}, \frac{5}{8})$. Projecting to our range from -1 to 1, this gives us a control estimate of $(-\frac{1}{4}, \frac{9}{16}, \frac{5}{8}, \frac{9}{16}, -\frac{1}{4})$.
This time it is more than twice as likely for white to control the points adjacent to the end than the points one step in. We find that the total (sum of) control over the empty space is $\frac{5}{4}$ which is just slightly in black's favour. This is not overwhelming given that black has 10 stones adjacent to the empty area while white only has 2. We can see how difficult it is to surround territory.

Figure 4: Doubly open-ended corridor

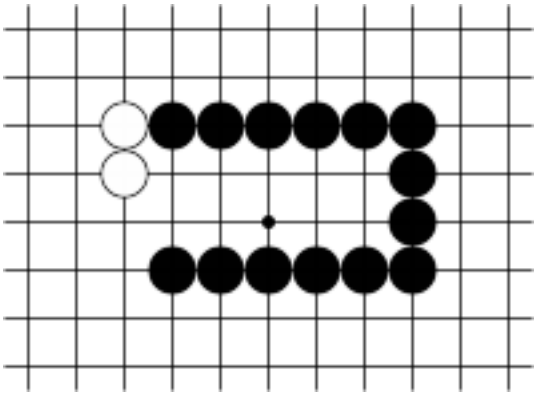
side plays the initial move doesn't matter, because the other side always has the option to return to an even result by responding. Such exchanges can be ignored in the game tree because they don't change the local score. Note that endgame theory shows that in the technical sense there is no such thing as double sente and a position can only be sente for one side and not the other. This makes evaluating which line to calculate unambiguous.

If you know a move is sente, you will have to slightly modify your estimate of score as in the above analysis on corridors. In the case of the width two corridor with a white stone at both entry points, if we didn't know the first white move was sente, we would have made the estimate that if white plays first, white penetrates $\frac{59}{16}$ points (double that of when there is only one reducing white stone as the variations are the same), and if black plays first, white still penetrates 1 point. This is an average of $\frac{75}{32}$, which is higher than the correct value of 2 points. The difference is because white's initial move doesn't make so much of a difference compared to black's response[5] , so black gains by assuming black will respond rather than leaving a 50-50 chance of either side taking the next move. In general, if a move is found to be sente, the new score taking this into account will be better for the side that is forced to respond, compared to the original calculation for the miai value. This may sound counter-intuitive, but for example, in the above corridor example, this is because the miai value calculation has already given black a lower score if black plays first in that position, so the calculation assuming white's first move is sente only has to be better for black than a score that has already been lowered.

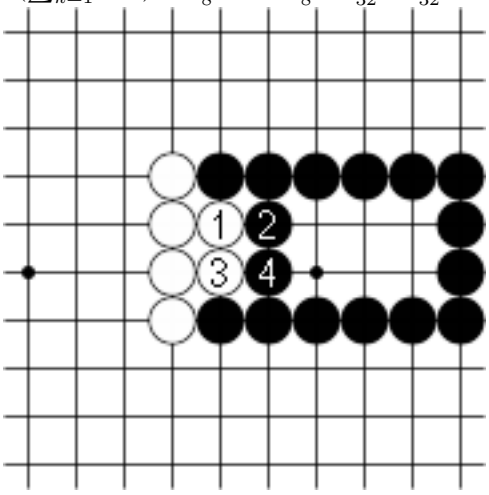
By adding some reading into your analysis, you could feed a control estimate into producing an estimate of miai value by entering possible moves and using the score estimate to work out the difference between the score if black plays first and if white plays first. Because there are generally lots of endgames on the board and using your initiative to take one big move allows your opponent to take the next biggest, endgame theory says you generally have to double this difference to estimate the miai value of the largest move on the board. If you know that in a local position black playing A is sente, forcing white to respond at B, then you can modify your control estimate accordingly. Of course, reading such sequences is very costly in computation time, so I wonder if there are simple heuristics that help.

Standard examples of sente moves include threatening to save or kill a larger group of stones, or threatening to

[5]Crudely we can say it is worth less. But technically, that's incorrect because it is a valuation based on not assuming the (sente) move is sente.



In this shape, black can make 8 points by playing first and valuation for the other possibilies if black responds after white pushes a 1,2,3,4 times are $6, 4, 17/8, 0$ points. The 17/8 is the expected score for a length 4 corridor. It is slightly bumped up by $\frac{1}{8}$ from 2 because black has an extra option of ignoring all of white's incursions from that point on. In any case, the moves get smaller so no move in this calculation is sente. White can penetrate $2(\sum_{k=1}^{4} 2^{-k}) - 2\frac{1}{8}2^{-3} = \frac{15}{8} - \frac{1}{32} = \frac{59}{32}$



Move 1 threatens to reduce by a further 2 points each time making the moves a nearly 4 point gote if it is counted as gote. In contrast if black had played first in the position, black would only get 1 point more in reverse sente (comparable to 2 point gote), so that doesn't compare. White's first move is sente, and the position is expected to settle to 6 points (± 1 as 3 is not sente so can't be assumed to be played next).
White's influence penetrates 2 points. So the extra supporting stones make almost no difference to black's territory.

Figure 5: Width two corridor

penetrate into or seal off a larger territory. An example suggested in GnuGo's documentation is peeping at a one space jump. Although connecting against a peep is normal, there are sometimes better moves such as an active counter-peep. Furthermore, it is awkward to statically assume that a peep for connection exchange will happen because that discards the possibility that the player might directly cut. This requires reading.

2.7 Influence functions vs Control Estimates

So far I have been talking about influence functions and control estimates interchangeably but they follow quite different principles. Influence functions measure the effect of stones on the rest of the game, following principles such as not passing through strong groups. Control estimates measure the probability that at the end of the game a given intersection will be controlled by black or white.

In particular, stones that are loosely surrounded in a net will count as dead in a control estimate because at the end of the game, they will most likely be controlled by the capturer. However, for the purposes of influence on the rest of the game (between now and the end), any stone still on the board will exert an influence on the board via the neighbouring intersections, so their presence can still affect the game. Having a few more stones of your colour on the board, even if they are going to be captured in this way, is normally still a bonus.

Katago trains to give a control estimate. And although endgame analysis is based on a control estimate, iterative methods discussed in this article mimic the propagation of influence functions and share many of their properties. In order for stones captured in a net to still make a positive effect on the score estimate from such methods it is natural to not count them as completely dead and to still consider their influence. Similarly a ponnuki that isn't yet connected into an eye may have one of the 4 stones around it captured by the opponent later, so it would be sensible not to immediately consider the space inside immediately controlled by the capturer unless all the 4 stones around it are strong.

If you sum the values for a control estimate, you should get an estimate for the score (before komi) by assuming the estimates for difference intersections have an error that is independent of each other. But I understand Katago has an independent output for the score estimate to be more accurate.

3 Symmetric ideas

3.1 Diagrams

kataestimate: we see that the system is way off in terms of accuracy of estimate.

Consider Figures 6-9. The score estimates are obtained by summing the values for the control estimate over the whole board. The number of iterations is shown. It was run until the norm squared of the difference accrued on an update step was less than the number in brackets, generally 1e-4.

Figure 6 assumes all stones on the board are alive, fixing the estimate on those intersections as -1 or 1 accordingly to their colour. On the empty intersections, the iterative procedure updates each cell s_i according to the function:

$$s_i = f(a_{i-1}, b_{i-1}, c_{i-1}, d_{i-1})$$

where a, b, c, d represent the estimates on the neighbouring intersections, and the subscript i denotes the ith iteration.

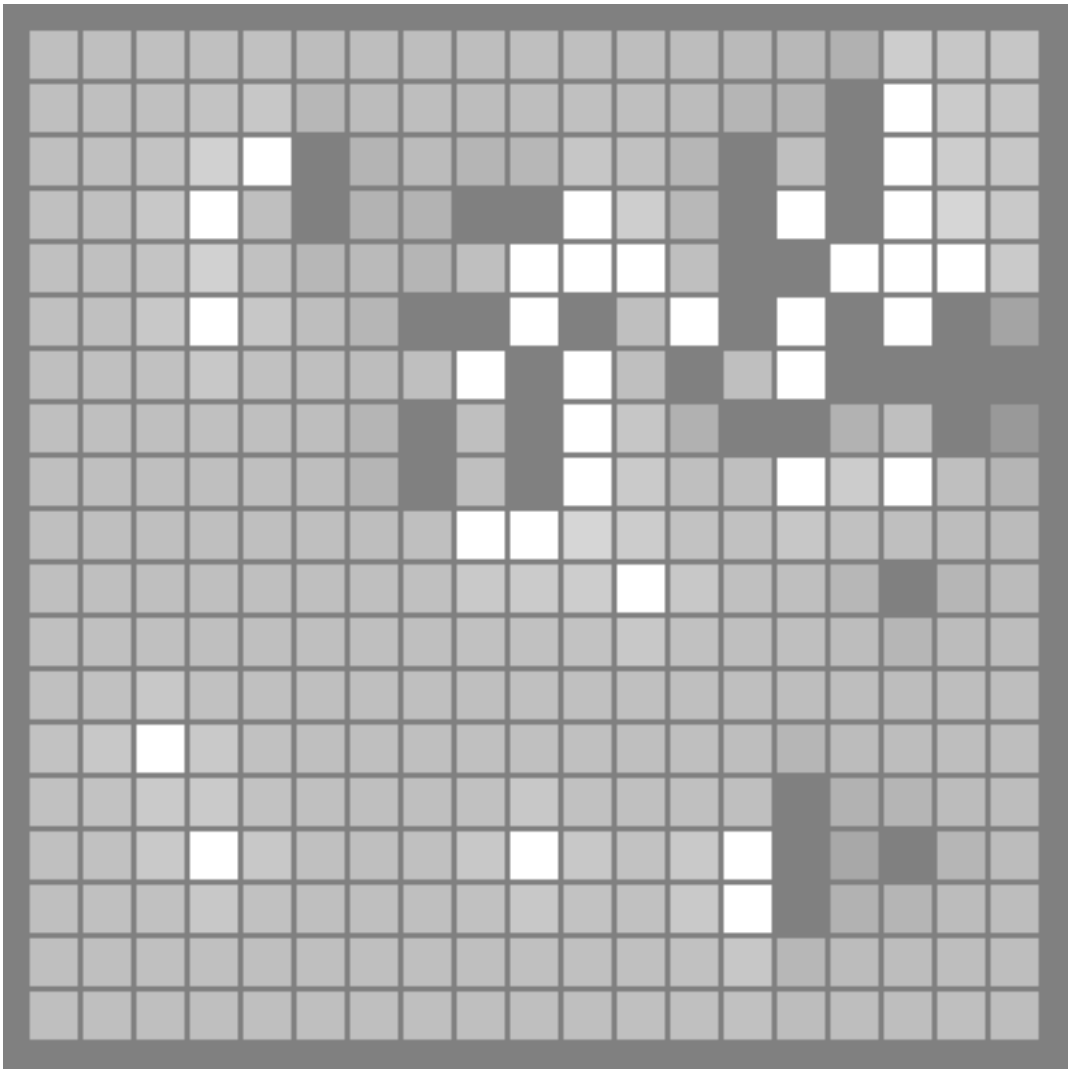
$$f(a,b,c,d) = \frac{1}{16}\left[(1+a)(1+b)(1+c)(1+d) - (1-a)(1-b)(1-c)(1-d)\right]$$

$$= \frac{1}{8}\left[a+b+c+d+bcd+acd+abd+abc\right]$$

This is somewhat of a smoothing function between areas controlled by white and those controlled by black. When intersections are on the edge of the board, the update function is the same sort of symmetric function. For example, in the corner, $f(a,b) = \frac{1}{4}[(1+a)(1+b) - (1-a)(1-b)]$.

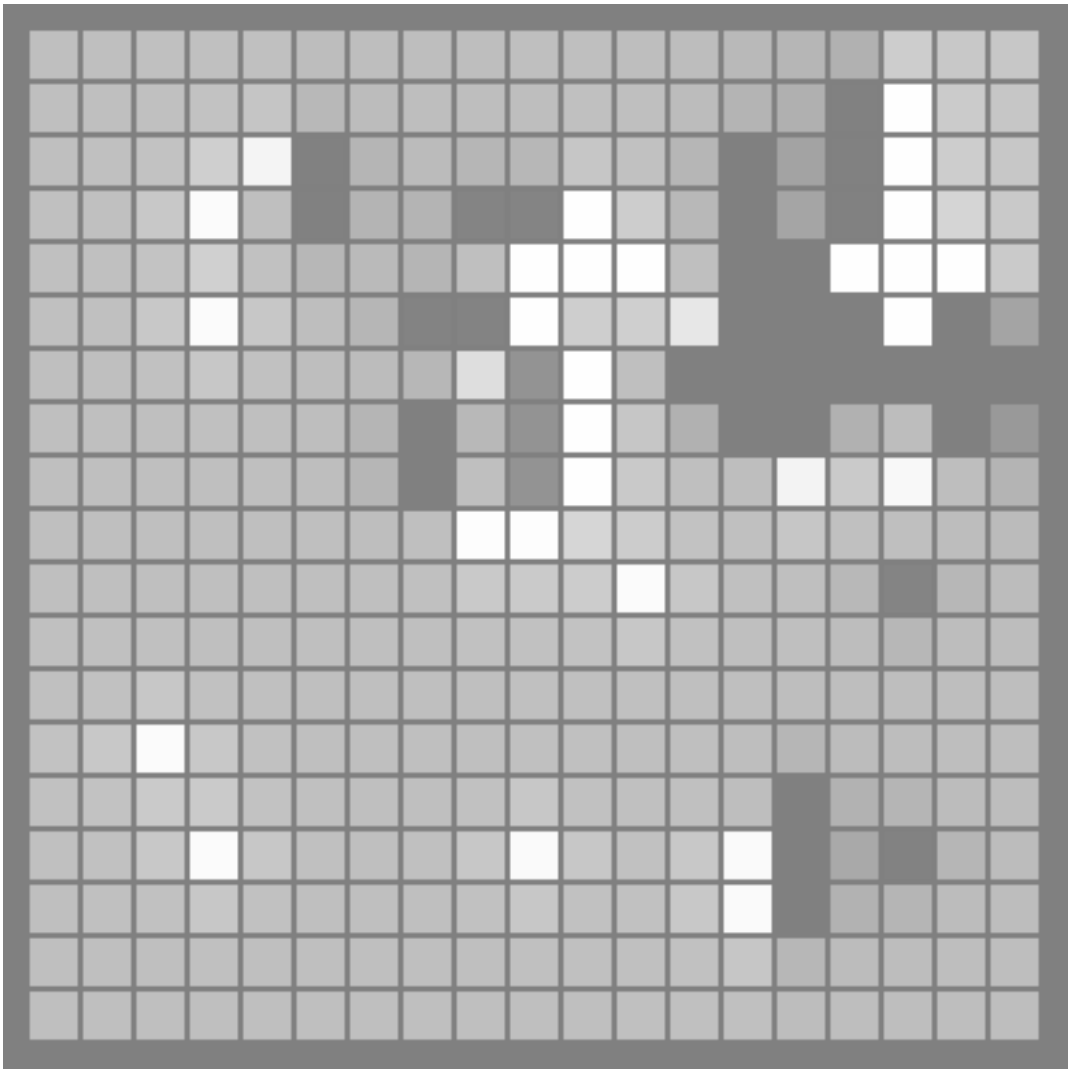
Figure 7 modifies the function for living stones to depend on the estimates in all the neighbouring cells. For a black chain, the update procedure is

$$s_i = 1 - 2\prod(1 - a_{i-1})/2$$



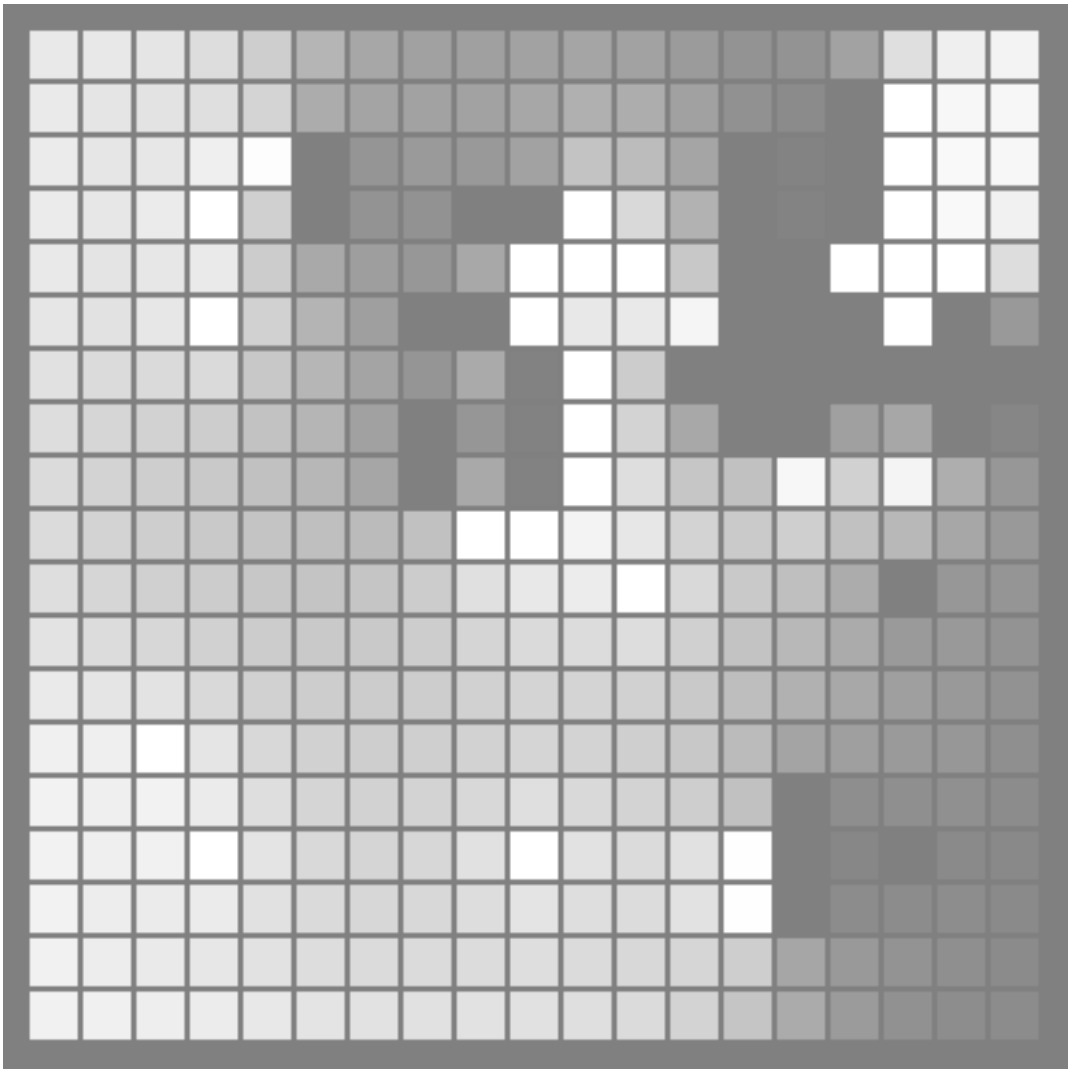
W+0.11
3 iterations (1e-3)

Figure 6: Analysis: All stones alive



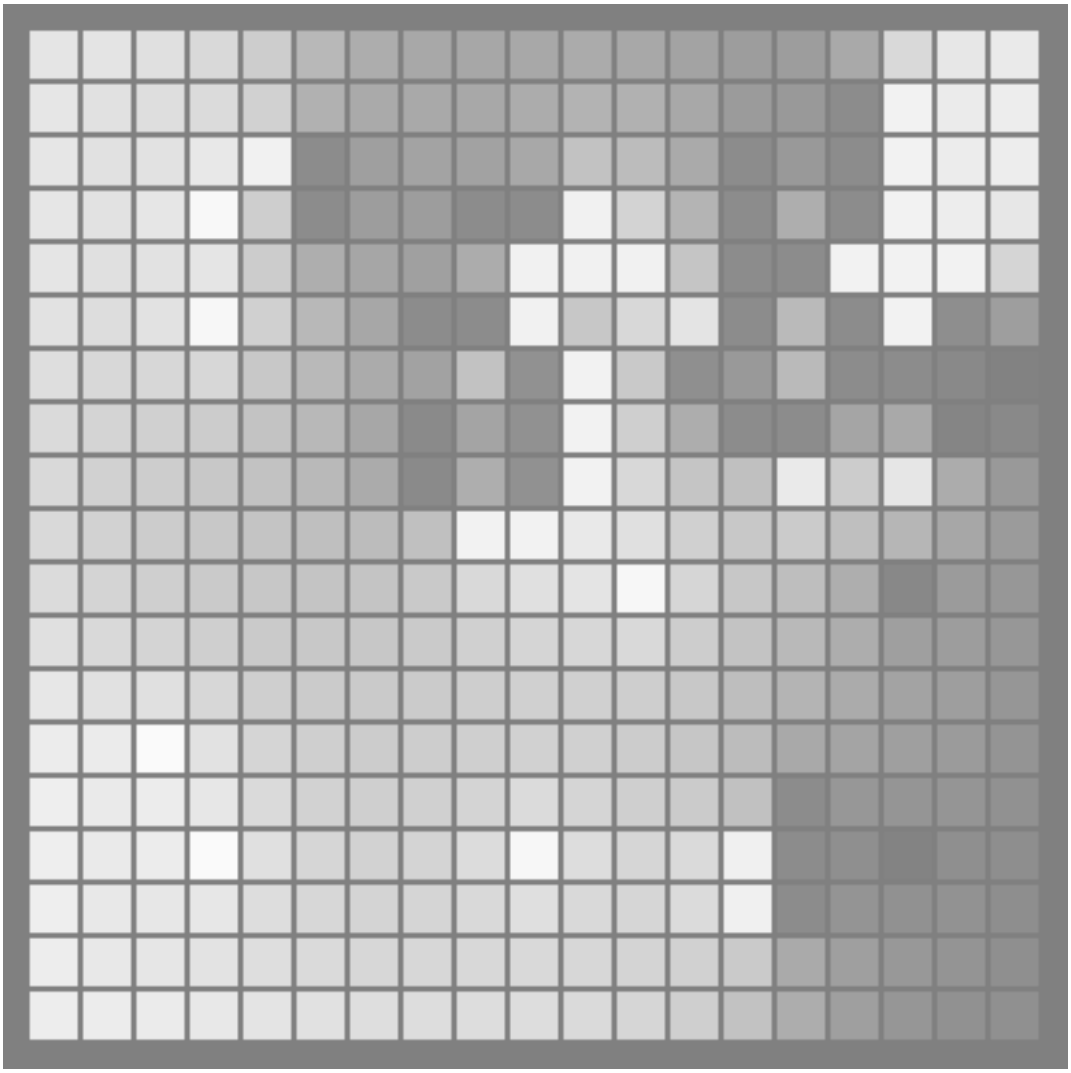
B+6.65
7 iterations (1e-4)

Figure 7: Analysis: Stones aliveness judged by neighbours



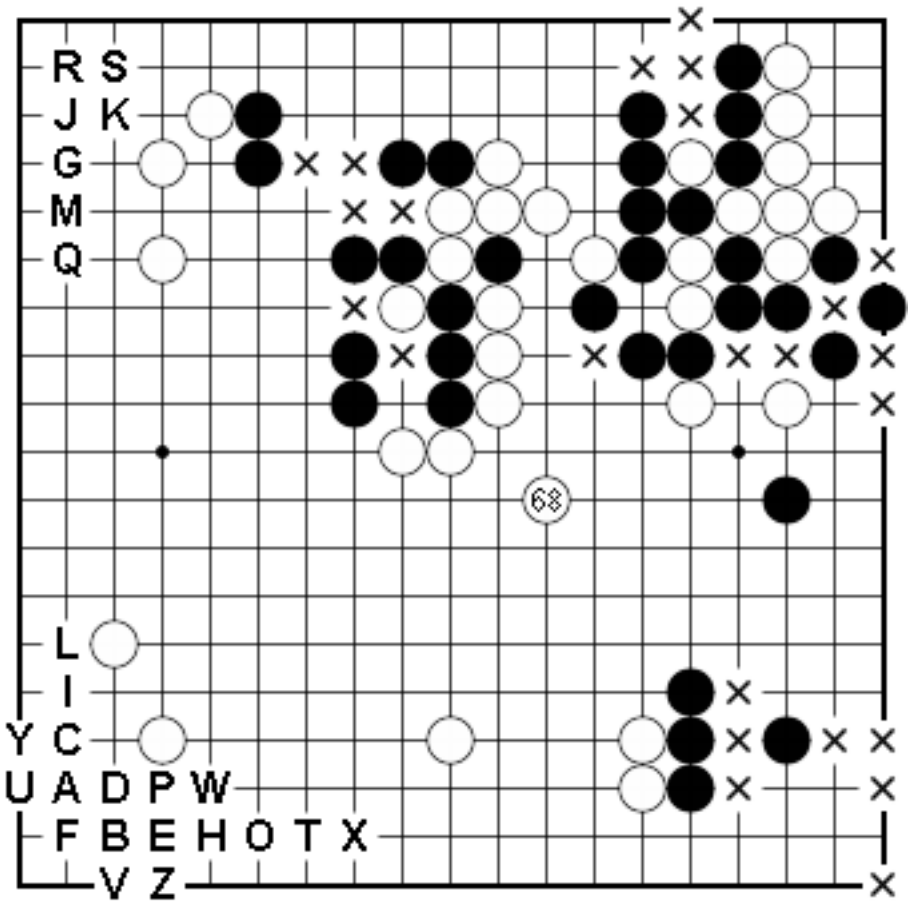
W+26.48
45 iterations (1e-4)

Figure 8: Analysis: Set $\lambda = 0.44$



W+27.03
48 iterations (1e-4)

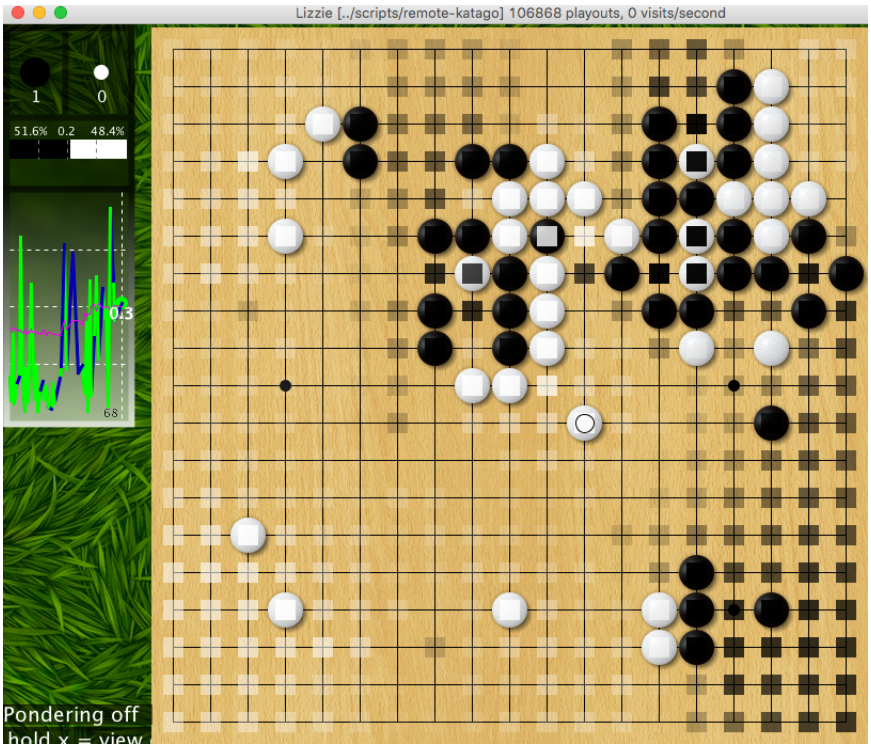
Figure 9: Analysis: Set $\kappa = 0.8$



This lists the top 26 moves in order by letters and marks the bottom 26 with a cross, according to this iterative system. This is computed by reading one move ahead and sorting the results of the score estimate, using $\lambda = 0.44, \kappa = 0.8$.

This system overestimates how alive stones on the board are or else it wouldn't suggest first line moves, but the direction of play of its suggestions at least makes sense. The bottom moves are overconcentrated moves that defend areas where black already has near complete control, other than the cutting point on the upper side.

Figure 10: Top and bottom moves



Computed at k=13 komi to minimise bias for 100k playouts, katago's score estimate is B+0.2, so B+13.2 overall on the board. But note that it is black to move, which probably gains around 7 points compared to a miai value estimate.

Katago shows a grayscale control estimate overlayed on the board position. It is much more confident about the corners and other territories. This shows a serious flaw in how my system overestimates moyos that have an invasion point. From this point of view, GnuGo's adjustment to add influence to invasion points not yet on the board makes sense. Also note there is a good chance the upper right is seki (i.e. 10 thousand year ko).

Figure 11: KataEstimate

where n is the number of adjacent intersections it has (i.e. empty or occupied by white stones), and the product is taken over the a values, the estimates on those intersections. The update for a white chain is the mirror image.

$$s_i = 2\prod(1 + a_{i-1})/2 - 1$$

The logic behind this is that if a represents a control estimate then $(1-a)/2$ scales it to represent a probability, of being controlled by white. So if the probability a black chain is dead is the probability that all the surrounding intersections are occupied by white stones, then we just take the product of $(1-a)/2$ to obtain the P, the probability of death (assuming independence of those probabilities). Finally, $p_i = 1-2P$ scales it back to the range from -1 and 1 to make it a control estimate. Of course this heuristic is very dodgy, not considering for example the possibility that even if black controls a neighbouring intersection, the whole connected chain might still be dead, for example in a connect and die situation. But the hope is that this is partially accounted for in the lambda modification if white has strength nearby.

I also considered using $p_i = 2^{-n}$ where n is the number of liberties but this seemed less flexible and relevant. Certainly if all the liberties are controlled by the opponent, the group should be counted as dead, but it isn't so clear what to do in the other cases. What we want to avoid is having a dead group that is counted as strong because it has a lot of liberties that are counted as strong simply because the group is counted strong. This merely forms a circular loop of logic. I wondered if a loop in a chain of stones could be made to count as eyespace but one of the many difficulties is that not all eyes occur from a loop. In any case, after having decided on a way to make stones a source of influence, the next problem is how to balance 1) ensuring influence steadily decreases so that one stone doesn't dominate the whole board with 2) ensuring that territory is considered territory.

Figure 8 introduces lambda. The update function for empty intersections is now

$$f(a,b,c,d) = \frac{1}{16}\left[(1+a)(1+b)(1+c)(1+d) - (1-a)(1-b)(1-c)(1-d) + \lambda h\right]$$

where

$$h = \sum_{\text{sym}}(1-a)(1+b)(1+c)(1+d) - (1+a)(1-b)(1-c)(1-d)$$

There are 8 terms obtained by total symmetry between a, b, c, d (equivalent to cyclic symmetry in this where only one is identified as special in each term).

$$= 2\left[a + b + c + d - bcd - acd - abd - abc\right]$$

Figure 9 introduces kappa. The update function for stones on the board is now a weighted average. It is κ times the previous function for a stone plus $1-\kappa$ times the previous function for an empty intersection. The idea is to reduce confidence in whether stones are alive or dead, especially when the opponent has strong stones nearby. One side effect turns out to be that stones captured in a net are now considered more alive than before as the stones around are considered less alive. This makes the estimate closer to an influence function rather than a control estimate. One might think this would skew the score incorrectly but it didn't reduce the score difference at all, and even increased it, perhaps because white's dead stones were considered more alive and white's main moyo in the lower left wasn't so negatively affected.

Finally note that if $\kappa = 0$, then the whole board is treated equivalently whether there is a stone there or not. In this case, the estimate quickly converges to the only stable solution of all zeros.

3.1.1 Lambda

For $0 < \lambda < 1$ the function f is increasing in all arguments, as only two decreasing terms balanced out. For example, just considering it as a function of a, first note that all the factors $(1 \pm a)$ are non-negative. There are only two terms decreasing in a when $\lambda > 0$ are (ignoring the 1/16 factor)

$$\lambda(1-a)(1+b)(1+c)(1+d)$$

and

$$-\lambda(1+a)(1-b)(1-c)(1-d).$$

By adding these to the first component, namely $(1+a)(1+b)(1+c)(1+d)-(1-a)(1-b)(1-c)(1-d)$, we obtain

$$(\lambda+1+(1-\lambda)a)(1-b)(1-c)(1-d)-(\lambda+1+(\lambda-1)a)(1-b)(1-c)(1-d).$$

This is still increasing in a for $\lambda<1$ and hence the whole update function is also increasing. We also have

$$f(1,1,1,1)=1, \quad f(-1,-1,-1,-1)=-1$$

so all this implies the update outputs are well defined results in the range $[-1,1]$. That is to say $f:[-1,1]^4\to[-1,1]$.

In the Figures, $\lambda=0.44$ was chosen because with one black stone on the board, it produced a score of around B+14, which is twice komi (we want a miai value half way between an even game and a two stone handicap game difference which are what result when white and black play next respectively).

When $\lambda=0$, f will only produce a positive result in black's favour if three of the estimates on neighbouring intersections move towards black's favour. If we consider fixing d, the first term responds to changes in d with a coefficient of $(1+a)(1+b)(1+c)/16$. We have $f(1,1,1,x)=(x+1)/2$ and $f(1,1,x,y)=(x+1)(y+1)/4$. These results may be more appropriate for corridor endgame situations. The contribution that λ makes is to add a term that contributes to black's favour even if only two of the estimates are in black's favour. This allows the influence to propagate much further in empty space. We now have

$$f(1,1,1,x)=\frac{\lambda+1+(1-\lambda)x}{2}$$

$$f(1,1,x,y)=\frac{1+x+y+xy+2\lambda(1-xy)}{4}$$

In particular when $\lambda=1/2$,

$$f(1,1,x,y)=\frac{2+x+y}{4}$$

This removes the higher order term which might be of some significance.

Although this doesn't fit the endgame results for corridors so well when lambda is added, there is some possible justification based on influence functions rather than control estimates. Because although these extra supporting black stones (or influence) might not increase the chance of controlling an area if the opponent is also strong there, but at least they will have an extra impact in reducing the liberties of an enemy stone that comes near. This counts for something, especially if propagated across the board.

3.1.2 Convergence

I am somewhat surprised and hence not that clear why these updates seem to always converge to some solution whatever board position I've tried, but the fact that there is a smoothing component to it makes it feel like the contraction mapping theorem is relevant to a proof, see the Appendix. This makes me think of why neural networks at Go keep improving from training and not (yet) go round in circles. It seems like for every step of progress made, this is relative to the last batch of self-play games, which could contain bias. Each training step adds knowledge but also may forget some older knowledge, so it seems hard at first sight to justify why the total knowledge increases relative to the whole game, and not just the last batch. Heuristically it seems to be to do with the large number of parameters in the networks so it is more likely to forget random structures than trained structures, and also because it is possible to match patterns with results in go and go positions aren't just random.

If we just consider $\lambda=0$, we have

$$f(x,x,x,x)=\frac{1}{16}[(1+x)^4-(1-x)^4]=\frac{1}{2}(x+x^3)=g(x)$$

The fixed points are at

$$x^3-x=0$$

or $x=1,0,-1$. So there are certainly the 3 constant solutions. As g is cubic, being convex on $[0,1]$ and concave on $[-1,0]$, we easily see that $x=0$ everywhere is the stable solution and $x=\pm1$ is unstable. This is all in the absence of influence sources such as stones on the board. At the least, we see that as we move away from sources, the estimate will tend to 0, representing $50-50$.

When $\lambda > 0$, we have

$$f(x,x,x,x) = \frac{1}{2}(x + x^3) + \lambda(x - x^3)$$

There is some sort of critical point at $\lambda = \frac{1}{2}$ when the cubic term changes sign and $f \equiv x$ with a fixed point everywhere. Convexity flips at this point. In this scenario, influence might propagate indefinitely. This helps explain why $\lambda = 0.44$ slightly below that was a good choice. As $\lambda \to 1$, the gradient of f goes to 0 at $x = \pm 1$ though f is still increasing.

3.1.3 Compute Speed

Without lambda, the computation is much faster, but lambda adds in an order of magnitude more iterations. At around 50 iterations, an analysis for just one board position takes nearly 3 seconds on my machine. Frankly this is very slow.

3.2 Heuristic assumptions and justifications

We can make some simplifying assumptions when we consider the estimate of a position based on the estimates of its neighbours. We suppose that the probabilities that each neighbouring position is controlled by black or white is independent. And also that if all neighbours are controlled by black, then that square must also be, and similarly for white. When not all controlled by black or white, we can consider various possibilities, the simplest of which is to divide the remaining probability $50 - 50$ which corresponds to $\lambda = 0$.

Any function should be the same across all intersections across the board, both because the rules of Go are that way, and also for simplicity. Furthermore, if the estimates for black controlling the neighbouring positions are a, b, c, d in anti-clockwise order, we expect the estimate for the central position $f(a, b, c, d)$ to have cyclic symmetry in a, b, c, d and maybe even total symmetry.

We know $f(1, 1, 1, 1) = 1$ and $f(-1, -1, -1, -1) = -1$. The simple function

$$f^* = (a + 1)(b + 1)(c + 1)(d + 1)/16 - (1 - a)(1 - b)(1 - c)(1 - d)/16$$

already satisfies all these conditions. So, assuming f can be expanded in these factors as a multinomial, we can conclude that our function f must be of the form $f = f^* + h$ where h is some cyclically symmetric combination of terms each of which includes at least one of the factors of $(a + 1)(b + 1)(c + 1)(d + 1)$ and one of factors of $(1 - a)(1 - b)(1 - c)(1 - d)$. This is by the factor theorem for multinomials.

It is also natural to assume the function is symmetric in black and white so that $f(-a, -b, -c, -d) = f(a, b, c, d)$. We next assume for simplicity that the function is homogeneous in those factors so that each term is degree 4, and also that f is linear in each of the arguments a, b, c, d, so that each factor appears exactly once in each term. This means that terms of the form $(a + 1)(b + 1)(1 - c)(1 - d)$ are eliminated and we need only consider the coupling coefficient of terms of the form $\lambda(a + 1)(b + 1)(c + 1)(1 - d)$. There are 4 terms of this form, and another 4 by the colour symmetry.

This is how we have obtained λ and justified our update function.

From the perspective of corridors, we might expect $f(1, 1, 1, x) = (1+x)/2$. On the other hand, even corridors are not simple enough to have an exact solution from this sort of local iterative method. This is because all corridors of different lengths have the same numbers for control estimate at the end where the invader is, but different numbers on the other end. Even if the iterative function took as argument the estimates of intersections at least two steps away, I don't think that would solve the problem.

On the other hand, without λ, we do have $f(1, 1, x, y) = (1 + x)(1 + y)/4$. For x, y close to 1, this is close to $(x + y)/2$. For a corridor open at one end we would like $f(1, 1, 1 - 2^{-n}, 1 - 2^{-(n+2)})$ to be $1 - 2^{-(n+1)}$. Our actual result is

$$f(1, 1, 1 - 2^{-n}, 1 - 2^{-(n+2)}) = 1 - 2^{-(n+1)} - 2^{-(n+3)} + 2^{-(2n+4)}$$

. The difference with 1 is off by a factor of nearly 1/4.

This difference could be interpreted by considering that our function doesn't distinguish between influence flowing in different directions. By assuming the neighbouring intersections are the sources, our function f predicts an estimate slightly closer to 0. However, in a corridor the influence only flows in one direction and invading stones near the entrance must be placed before invading further inside. Hence as the influence decreases with range, this should reduce the influence of the invader closer to the sink rather than in the middle.

3.3 Places to improve

For all this thought, the deficiencies are plain. It doesn't have much advantage over propagating functions other than being more confident about potential territory, a built-in estimate of when stones are alive or dead (but not very accurate unless nearly surrounded already), and the endgame problems the ideas were based on.

1. Life and death
2. Sente moves

A Convergence under smoothing operators

Admittedly I only have undergraduate level knowledge about this.

A.1 Averaging

The classic smoothing operator would update each cell with the average of its neighbours. So, considering the 1D problem, fixing the ends as boundary conditions, we can update

$$(a_0, a_1, \cdots, a_n) \to (a_0, \frac{a_0+a_2}{2}, \frac{a_1+a_3}{2}, \cdots, \frac{a_{n-2}+a_n}{2}, a_n)$$

We can try to construct a loss function. We are inspired by knowing that this should converge to a "line" which evenly goes from a_0 to a_n with $a_i \to \frac{(n-i)a_0+ia_n}{n}$. And a line is the geodesic in Euclidean space. Hence, the final solution minimises a loss function $L = \sum_{i=0}^{n-1}(a_{i+1}-a_i)^2$.

Now after one update step, this becomes $\left(\frac{a_0-a_2}{2}\right)^2 + \left(\frac{a_0-a_1+a_2-a_3}{2}\right)^2 + \cdots$. Writing $b_i = a_{i+1} - a_i$, we find $\sum b_i^2$ updates to

$$\left(\frac{b_0+b_1}{2}\right)^2 + \left(\frac{b_0+b_2}{2}\right)^2 + \cdots$$

Each b_i appears twice. This is truly less than or equal to $\sum b_i^2$ because 1) the sum of the elements we are squaring is the same, namely $\sum b_i$, 2) squaring is a convex function, so mixing the elements in the second case can only reduce the total compared to putting elements together according to size. Compare the rearrangement inequality.

Explicitly, we have

$$L_{\text{new}}\left(\frac{b_0+b_1}{2}\right)^2 + \left(\frac{b_0+b_2}{2}\right)^2 + \cdots \leq \frac{b_0^2+b_1^2}{2} + \frac{b_0^2+b_2^2}{2} + \cdots = L_{\text{old}}$$

by AM-GM or RMS-AM etc. which gives our desired result. This loss function decreases with each step, but as it is non-negative, it must get arbitrarily close to an equality case of the inequality. That is to say $L_{\text{new}} - L_{\text{old}}$ must tend towards zero. Equality occurs when all the b_i are equal, giving us our expected solution of a "line". Explicitly we have

$$L_{\text{new}} - L_{\text{old}} = -\frac{(b_0-b_1)^2}{2} - \frac{(b_0-b_2)^2}{2} - \cdots$$

This is related to the contraction mapping theorem though not equivalent as both construct a function of differences between one iteration and the next that must tend towards zero.

A.2 General cases

In general, how do we find such a loss function if it exists? I'm not sure. However, in the above case, we could view it in terms of the Lagrangian for a free particle treating the subscript i like time. Our update satisfies a second order difference equation with $2a_i - a_{i+1} - a_{i-1} = 0$ at a fixed point of the operator. This is analogous to $\ddot{x} = 0$ with the kinetic energy $\frac{1}{2}\dot{x}^2$ and no potential energy. The Lagrangian being $K - V = \frac{1}{2}\dot{x}^2$ as required. The Euler-Lagrange equation will then return $\ddot{x} = 0$.

One way to think about this, since we are dealing with a discrete problem, is that the derivative of the loss function at a point, say a_i, returns the "equation of motion", i.e. update operator at that point. The derivative of the loss function for the 1D average operator with respect to a_i is simply $2a_i - a_{i+1} - a_{i-1}$.

References

[1] Gnu Go Documentation

[2] Katago original paper

[3] The Alphazero paper

[4] Bill Spight's Thermography lecture