COMPUTER CHESS WITH MODEL PREDICTIVE CONTROL AND REINFORCEMENT LEARNING

Dimitri P. Bertsekas
Arizona State University

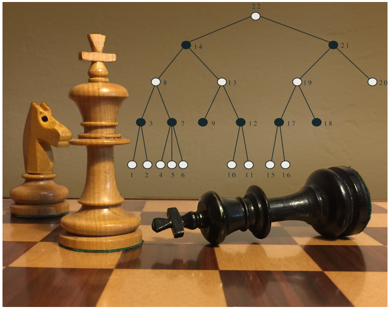
Lecture at Reinforcement Learning Workshop 2025 at the Indian Institute of Science, Bengaluru, January 2025

Based on my course at ASU on DP/RL (2019-2024), and my recent books
Lessons from AlphaZero for Optimal, Model Predictive, and Adaptive Control, 2022
A Course in Reinforcement Learning, 2nd Edition, 2025

Joint computer chess paper with Yuchao Li and Atharva Gundawar
"Superior Computer Chess with Model Predictive Control, Reinforcement Learning, and Rollout," arXiv:2409.06477, 2024

Outline

Computer Chess - AlphaZero (2017)

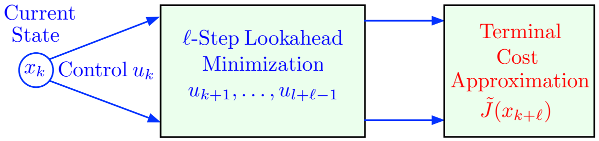


AlphaZero (and most chess programs) involve two algorithms:

- Off-line training of a position evaluator (among others), using deep NNs
- On-line play by multistep lookahead, and position evaluation at the end

- Most attention has been focused on the AlphaZero off-line training, which involves important innovations in NN technology, etc
- The on-line algorithm part is more or less traditional. It is critically important for good performance
- On-line play in computer chess is strongly connected with MPC

Model Predictive Control (MPC): Multistep Lookahead Optimization with Cost Approximation $\tilde{J}$ at the End



Current State
x_k
Control u_k
ℓ-Step Lookahead Minimization
$u_{k+1},\dots,u_{l+\ell-1}$
Terminal Cost Approximation
$\tilde{J}(x_{k+\ell})$

$\boxed{\text{At } x_k} \Longrightarrow$ Solve an ℓ-step lookahead problem with terminal cost $\tilde{J}$

1st ℓ Steps $\longleftrightarrow$ Future $\longleftrightarrow$

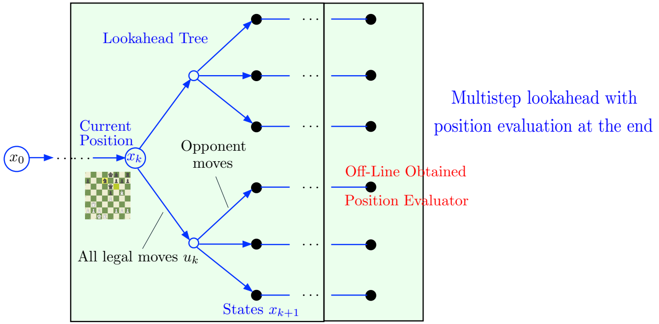
$$\min_{u_k,u_{k+1},\dots,u_{k+\ell-1}} \left\{ \sum_{m=0}^{\ell-1} g(x_{k+m},u_{k+m}) + \tilde{J}(x_{k+\ell}) \right\}$$

Apply the first control $\tilde{u}_k$, discard the remaining controls

Discrete-time deterministic optimal control problem

- Dynamic system equation at time k: $x_{k+1} = f(x_k, u_k)$
- State and control at time k: x_k and u_k
- Cost at stage k: $g(x_k, u_k)$

On-Line Play in Computer Chess


Lookahead Tree
Current Position
x_0
x_k
Opponent moves
All legal moves u_k
States x_{k+1}
Off-Line Obtained Position Evaluator
Multistep lookahead with position evaluation at the end

It is “isomorphic" to the MPC architecture, except:

- In chess the state and control spaces are discrete, while in MPC they are usually continuous
- In chess the lookahead tree is usually “pruned", while in MPC the lookahead optimization is usually exact (more on this later)
- The differences are inconsequential: Our Newton step theory allows arbitrary state and control spaces, and inexact lookahead
- Another difference: Chess is a two-player game. More on this later, but think of chess against a fixed opponent (this makes chess a one-player game)

Principal Viewpoints of this Talk

- On-line play w/ one-step lookahead is a single step of Newton's method for solving the problem's Bellman equation (similar interpretation applies to multistep lookahead)
- Off-line training provides the starting point for the Newton step
- On-line play is the real workhorse ... off-line training plays a secondary role
- The Newton step framework is very general, because it is couched on abstract DP ideas (arbitrary state and control spaces, stochastic, deterministic, hybrid systems, multiagent systems, minimax, finite and infinite horizon, discrete optimization)

We focus on a meta-chess architecture that has yielded surprisingly favorable results

Visualization of MPC for Linear-Quadratic Problems

We consider one-dimensional problems for easy visualization (x_k, u_k: scalars)

- System: $x_{k+1} = ax_k + bu_k$, where a, b are given scalars
- Cost: $\sum_{k=0}^{\infty}(qx_k^2 + ru_k^2)$, where q, r are positive scalars
- Basic facts: The optimal cost function is a quadratic function of the state x, and the optimal control policy is a linear function of x

Optimal solution

- Optimal cost function: $J^*(x) = K^*x^2$ where K^* is the unique positive solution of the Riccati equation

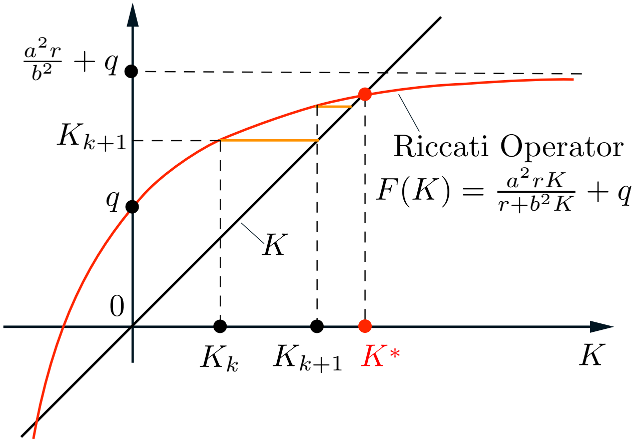
$$K = F(K), \quad \text{where} \quad F(K) = \frac{a^2rK}{r + b^2K} + q \quad \text{is the Riccati operator}$$

- Optimal policy: Linear of the form $\mu^*(x) = L^*x$, where L^* is the scalar given by

$$L^* = -\frac{abK^*}{r + b^2K^*}$$

The insights from one dimensional/linear-quadratic problems generalize

Graphical Solution of Riccati Equation and Value Iteration (VI)

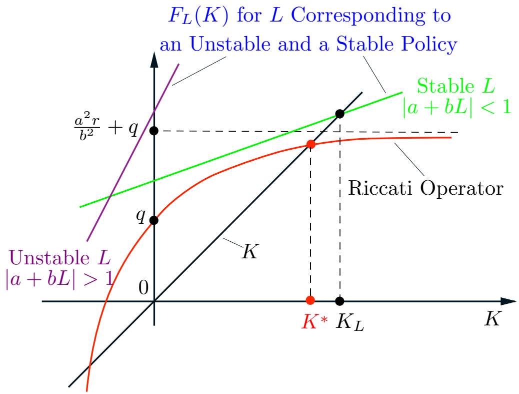


VI generates iteratively the optimal cost functions $J_k(x_k)$ of k-stage problems

J_k is quadratic of the form $J_k(x_k) = K_k x_k^2$, where $\{K_k\}$ is obtained by iterating with the Riccati operator F:

$$K_{k+1} = F(K_k), \quad k = 0, 1, \ldots, \qquad K_0 : \text{given}$$

Visualization of Riccati Equation for Linear Policies



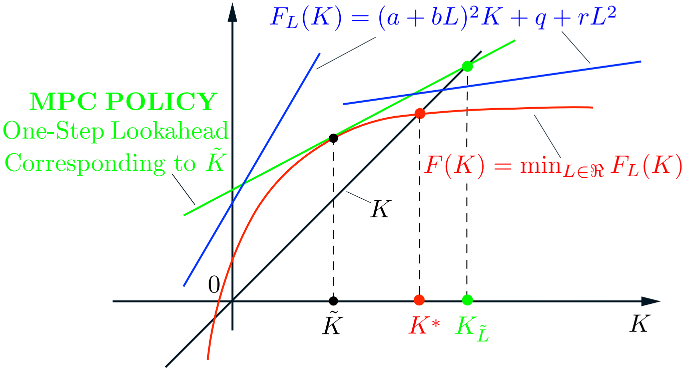
Consider a linear policy $\mu_L(x) = Lx$ and its cost function

It is quadratic of the form $K_L x^2$, where K_L is the unique solution of the Riccati equation for linear policies (also called Lyapunov equation):

$$K = F_L(K), \quad \text{where} \quad F_L(K) = (a + bL)^2 K + q + rL^2, \quad \text{it is linear in } K$$

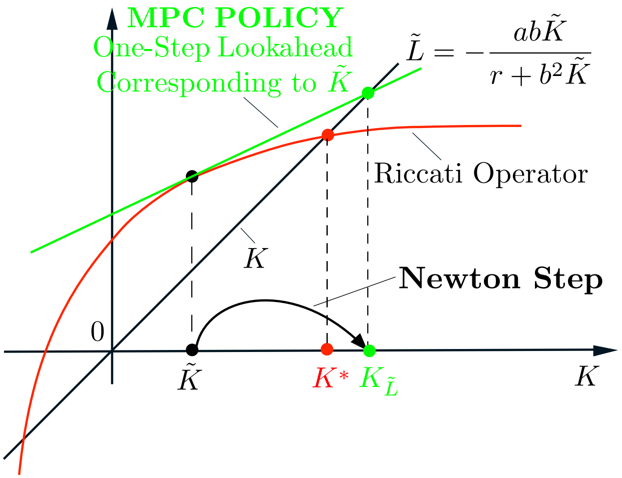
This is only for stable policies (those with $|a + bL| < 1$). For unstable policies $K_L = \infty$

Visualization of MPC (One-Step Lookahead)


$F_L(K) = (a+bL)^2K + q + rL^2$
MPC POLICY
One-Step Lookahead
Corresponding to $\tilde{K}$
$F(K) = \min_{L\in\Re} F_L(K)$
K
0
$\tilde{K}$
K^*
$K_{\tilde{L}}$
K

- The graph of F is the lower envelope of the lines corresponding to linear policies (stable as well as unstable)
- The tangent line corresponding to $\tilde{K}$ defines the (one-step lookahead) MPC policy with terminal cost $\tilde{K}x^2$
- It linearizes the Riccati operator at $\tilde{K}$
- Linearization is a critical property for the Newton step interpretation of MPC (concavity of the Riccati operator is also important)

Newton Step Visualization

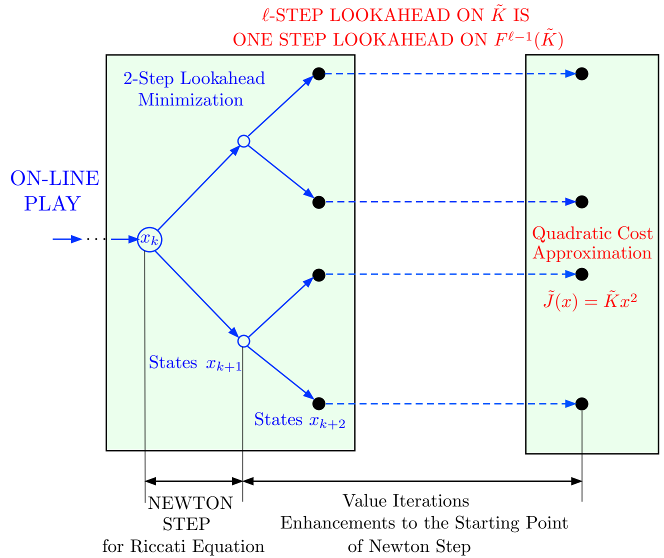


The meaning of superlinear convergence:

The error $|K_{\tilde{L}} - K^*|$ is MUCH smaller than $|\tilde{K} - K^*|$

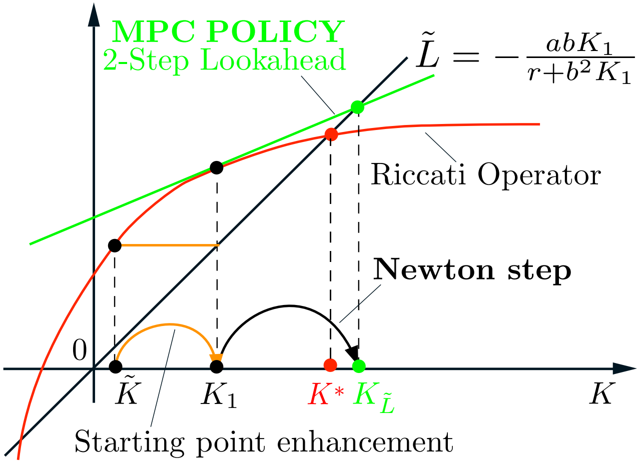
Explains the good performance of MPC in practice

Multistep Lookahead - Preview of the Newton Step


ℓ-STEP LOOKAHEAD ON $\tilde{K}$ IS
ONE STEP LOOKAHEAD ON $F^{\ell-1}(\tilde{K})$
2-Step Lookahead Minimization
ON-LINE PLAY
x_k
States x_{k+1}
States x_{k+2}
Quadratic Cost Approximation
$\tilde{J}(x) = \tilde{K}x^2$
NEWTON STEP for Riccati Equation
Value Iterations
Enhancements to the Starting Point of Newton Step

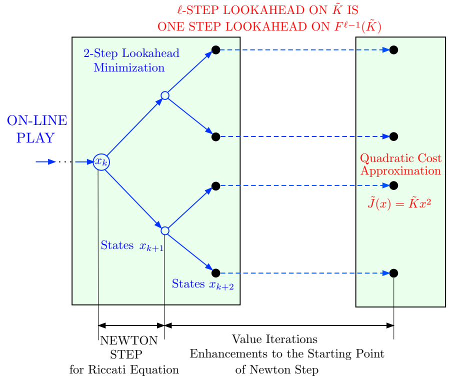
Only the first step of the lookahead is a Newton step

Multistep Lookahead - Illustration of the Newton Step



Multistep lookahead brings the starting point of the Newton step closer to K^*

The Importance of the First Step of Lookahead



ℓ-STEP LOOKAHEAD ON $\tilde{K}$ IS
ONE STEP LOOKAHEAD ON $F^{\ell-1}(\tilde{K})$
2-Step Lookahead Minimization
ON-LINE PLAY
x_k
States x_{k+1}
States x_{k+2}
Quadratic Cost Approximation
$\tilde{J}(x) = \tilde{K}x^2$
NEWTON STEP for Riccati Equation
Value Iterations Enhancements to the Starting Point of Newton Step

- In ℓ-step lookahead MPC, only the first step of lookahead acts as a Newton step
- The remaining $\ell - 1$ steps only serve to enhance the starting point of the first/Newton step
- Important insight: The first minimization step should be done exactly, the remaining steps can be done approximately
- Application: In stochastic problems, use certainty equivalence approximations in all lookahead steps except the first (Bertsekas and Castanon, 1999)

Range of Applications of MPC/Rollout - Our Work at ASU

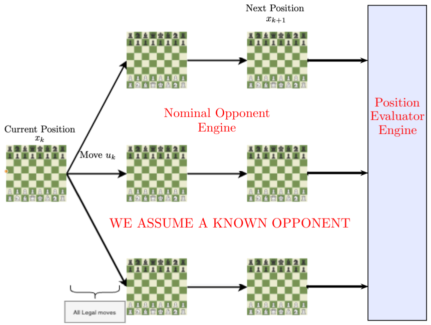
We have considered applications involving discrete state and control spaces

- Discrete/combinatorial optimization (e.g., scheduling problems)
- Multiagent robotics: Maintenance/repair, taxi fleet management - joint work with Stephanie Gil's group at Harvard (2021-2023)
- Data association and multitarget tracking - Musunuru, Li, Weber, and DPB, 2024
- Sequential inference/decoding problems and the Wordle puzzle - Bhambri, Bhatacharjee, and DPB, 2023)
- Most likely sequence generation in ChatGPT-like transformers, related HHM inference, and Viterbi-rollout algorithm - Li and DPB, 2024

We will now focus on computer chess

Ref: ArXiv paper by Gundawar, Li, and DPB, 2024

MPC-MC (MetaChess): An Algorithm for Better Computer Chess



We introduce a one-player MPC architecture (the true opponent is approximated by a “nominal" opponent)

We use two available chess engines as components (a meta algorithm)

- The nominal opponent engine: Predicts the move of the true opponent of MPC-MC (exactly or approximately)
- The position evaluator engine: The base engine; evaluates any given position
- Each move involves a Newton step starting at the position evaluation function

MPC-MC: Computational Results Using the Stockfish (SK) Base Engine

We tested two variants of the algorithm

- Standard version
- Fortified version (plays a little better against very strong opponents, a little worse against weaker opponents)

Table: MPC-MC vs SK

SK Strength	Exact. Known Opponent		Approx. Known Opponent	
	Standard	Fortified	Standard	Fortified
0.5 secs	7.5-2.5	8-2	8-2	7-3
2 secs	5-5	5.5-4.5	5.5-4.5	6.5-3.5
5 secs	5-5	5.5-4.5	10-10	10.5-9.5

- We use MPC-MC (one-step lookahead), with SK as both the position evaluator and the nominal opponent, to play against SK
- Similar (but better) results obtained with other engines
- We can obtain better results with multistep lookahead
- Parallel computation is essential to reduce the move generation time

Our Speculation on Performance Improvement

Consistent observation:

MPC-MC improves the play of weak base engines (decisively), and the play of strong/world champion base engines (narrowly)

Why is this happening? (After all MPC-MC's lookahead is only one step longer)

- Observation: MPC-MC plays mostly the same moves as the base engine (SK) ... but varies in about 10-20 percent of the moves
- Occasionally, the base engine misses some hard-to-find strong moves at the first step of lookahead, which MPC-MC does not.
- The first step of lookahead is exact in MPC-MC. So it performs a true Newton step.
- This is all counterintuitive. However, it is consistent with the Newton step theory, and with much experimentation from other RL and MPC case studies.

Thank you!