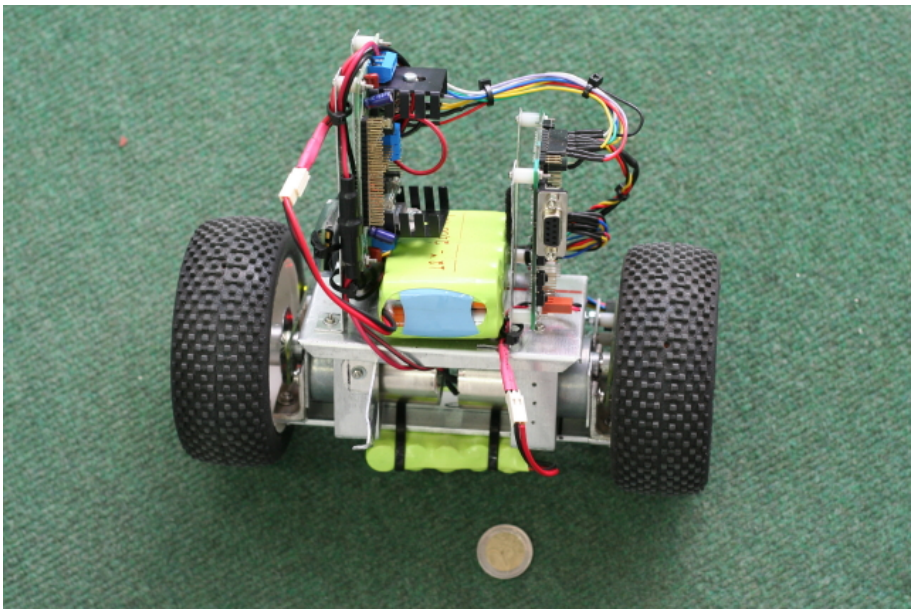
Μελέτη και Υλοποίηση Ελεγκτών Ρομποτικών Συστημάτων με χρήση Αλγορίθμων Ενισχυτικής Μάθησης



Κόντες Γεώργιος

Μελέτη και Υλοποίηση Ελεγκτών Ρομποτικών Συστημάτων με χρήση Αλγορίθμων Ενισχυτικής Μάθησης

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Κόντες Γεώργιος

Α.Μ: 2007019016, Email: gkontes@isc.tuc.gr

Εξεταστική Επιτροπή:

Ν. Βλάσσης, Επίκουρος Καθηγητής, Τμήμα Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης (Επιβλέπων)

Μ. Λαγουδάκης, Επίκουρος Καθηγητής, Τμήμα Ηλεκτρονικών Μηχανικών και Μηχανικών Υπολογιστών, Πολυτεχνείο Κρήτης

Β. Κουικόγλου, Καθηγητής, Τμήμα Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ
ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Εικόνα εξωφύλλου: RobBa

Μελέτη και Υλοποίηση Ελεγκτών Ρομποτικών Συστημάτων με χρήση Αλγορίθμων Ενισχυτικής Μάθησης

Περίληψη

Η παρούσα εργασία μελετά τις ιδιότητες αλγορίθμων ενισχυτικής μάθησης μέσω πιθανοτικού συμπερασμού. Προτείνουμε έναν νέο Monte Carlo Expectation - Maximization αλγόριθμο στο συγκεκριμένο πεδίο και συγκρίνουμε την ταχύτητα και την ποιότητα σύγκλισής του σε σχέση με υπάρχοντες αλγορίθμους. Στη συνέχεια προτείνουμε δύο μεθόδους επιτάχυνσης της συγκεκριμένης κατηγορίας αλγορίθμων, τη μέθοδο αναζήτησης επί γραμμής και τη μέθοδο χρήσης των καλύτερων μόνο τροχιών. Επιδεικνύουμε την αποτελεσματικότητα των αλγορίθμων και των μεθόδων επιτάχυνσής τους σε ένα διδιάστατο συνθετικό πρόβλημα, σε ένα πρόβλημα εξομοίωσης αιώρησης ελικοπτέρου και σε ένα πρόβλημα εύρεσης κατάλληλου ελεγκτή για την ισορροπία μιας δίτροχης ρομποτικής ιδιοκατασκευής.

Ευχαριστίες

Πρώτα απ' όλα θα ήθελα να ευχαριστήσω τον καθηγητή μου Νίκο Βλάσση. Χωρίς την υπομονή του και τις πολύτιμες συμβουλές του, αυτή η εργασία δεν θα είχε πραγματοποιηθεί. Από το Νίκο έμαθα να σκέφτομαι έξω από τα δεδομένα πλαίσια, να εργάζομαι αυτόνομα και πάντα να αναζητώ το κάτι παραπάνω και από τον εαυτό μου και από την εργασία μου. Ένα από τα μεγαλύτερα εφόδια στην διαδρομή μου από εδώ και πέρα θα είναι η μεθοδικότητα και ο τρόπος σκέψης που μου μετέδωσε. Θα μου λείψουν οι ατελείωτες συζητήσεις μας και το "brainstorming" καθ' όλη τη διάρκεια του μεταπτυχιακού μου, καθώς και ο σεβασμός του Νίκου προς όλους τους φοιτητές του και ειδικά προς εμένα.

Επίσης, θα ήθελα να ευχαριστήσω όλους τους φίλους που απέκτησα κατά τη διάρκεια των μεταπτυχιακών μου σπουδών, αλλά και τους φίλους από παλιά, και ειδικά το Γιώργο, για την συμπαράστασή τους και τις συμβουλές τους.

Ένα ξεχωριστό ευχαριστώ στον Στέφανο και στον Walid (Γουαλίκιο), τους "συγκατοίκους" μου στο εργαστήριο. Με τις συζητήσεις και το χαβαλέ τους βοήθησαν να περάσουν ευχάριστα οι ατελείωτες ώρες δουλειάς.

Ένα πολύ μεγάλο ευχαριστώ οφείλω στο Σάββα Πιπερίδη, ο οποίος κατασκεύασε τη RobBa και ήταν πάντα πρόθυμος να λύσει όλες τις απορίες μου. Χωρίς την εμπειρία και τις συμβουλές του Σάββα, το τεχνικό κομμάτι της εργασίας δεν θα είχε ολοκληρωθεί.

Στο σημείο αυτό θα ήθελα να ευχαριστήσω την οικογένειά μου, για την ηθική και υλική υποστήριξή τους όλα αυτά τα χρόνια. Χωρίς την εμπιστοσύνη και την υπομονή τους δεν θα είχα ολοκληρώσει τις σπουδές μου.

Τέλος, ένα μεγάλο ευχαριστώ στη Ναταλία που ήταν πάντα δίπλα μου...

Εισαγωγή

Η ιδέα να δημιουργήσουν οι άνθρωποι μηχανές που να τους υπηρετούν, να τους διασκεδάζουν, να τους βοηθούν ή να τους αντικαθιστούν στις εργασίες τους χάνεται βαθιά στο χρόνο, καθώς ήδη από το 400 π.Χ περίπου έχουμε αναφορές για μηχανές που λειτουργούσαν αυτόνομα (αυτόματα), αλλά έπρεπε να περιμένουμε μέχρι το 1920, οπότε ο Τσέχος συγγραφέας Karel Capek εισάγει τη λέξη ρομπότ στο θεατρικό του έργο R.U.R. (Rossum's Universal Robots) και εν αγνοία του δίνει το όνομά της σε μια από τις πιο ενεργές ερευνητικές περιοχές της σύγχρονης επιστήμης, τη ρομποτική.

Η ρομποτική σαν πεδίο έρευνας ξεκίνησε από απλούς ρομποτικούς βραχίονες και σήμερα έχει να "επιδείξει" μια πληθώρα διαφορετικών τύπων ρομπότ, όπως είναι τα βιομηχανικά, τα ιπτάμενα, τα υποθαλάσσια, τα οικιακά και πολλά άλλα. Το επόμενο βήμα είναι τα "έξυπνα ρομπότ", δηλαδή τα ρομπότ εκείνα που θα λειτουργούν αυτόνομα και θα μαθαίνουν από τα λάθη τους.

Οι κλασικές μέθοδοι που προσπαθούν να εκπαιδεύσουν ρομπότ να εκτελούν εργασίες προϋποθέτουν τέλεια γνώση της εργασίας που θα εκτελεστεί, αλλά και πλήρη ή μερική γνώση του περιβάλλοντος στο οποίο εκτελείται η εργασία. Γίνονται ήδη αντιληπτοί οι περιορισμοί που θέτουν αυτές οι προσεγγίσεις· κανείς δεν μπορεί να προβλέψει τη δομή και όλες τις συνιστώσες ενός δυναμικά μεταβαλλόμενου συστήματος, όπως είναι το πραγματικό περιβάλλον. Δεν μπορούμε να ελέγξουμε την ένταση και την κατεύθυνση των ρευμάτων αέρα που θα συναντήσει ένα αυτόνομο ελικόπτερο κατά την πτήση του ή να ενσωματώσουμε πληροφορίες σε ένα αυτόνομο ρομποτικό όχημα που εξερευνά κάποιον πλανήτη για την μορφολογία του εδάφους που θα συναντήσει. Πρέπει λοιπόν να σχεδιάζουμε ρομπότ που να μπορούν να αντιμετωπίσουν το άγνωστο και το απροσδόκητο, αλλά και να μαθαίνουν από αυτό.

Ένα πεδίο που ασχολείται ενεργά με τη μάθηση αυτόνομων πρακτόρων είναι η Ενισχυτική Μάθηση (Reinforcement Learning). Εδώ η διαδικασία της μάθησης είναι αυτόνομη· δεν απαιτείται κάποιος δάσκαλος να διδάξει στο ρομπότ πώς πρέπει να αντιδρά σε κάθε περίπτωση, αλλά το ίδιο το ρομπότ μέσα από την αλληλεπίδρασή του με το περιβάλλον μαθαίνει ποιες ενέργειες είναι προς όφελός του και ποιες εις βάρος του, ακολουθώντας ακριβώς την ίδια διαδικασία που ακολουθούμε όταν σαν παιδιά μαθαίνουμε να περπατάμε ή να κάνουμε ποδήλατο.

Στο δεύτερο κεφάλαιο θα παραθέσουμε τις βασικές αρχές της Ενισχυτικής Μάθησης και θα δούμε τις δύο επικρατέστερες τάσεις που επικρατούν στη σχεδίαση τέτοιων αλγορίθμων, τους αλγορίθμους που χρησιμοποιούν συναρτήσεις αξιολόγησης (value functions) και τους αλγορίθμους πολιτικής κλίσης (policy gradient). Στο τρίτο κεφάλαιο θα εξετάσουμε τις ιδιότητες δύο αλγορίθμων οι

οποίοι ανάγουν το πρόβλημα της Ενισχυτικής Μάθησης σε ένα πρόβλημα Πιθανοτικού Συμπερασμού (Probabilistic Inference) και το επιλύουν με τη βοήθεια του αλγορίθμου Expectation - Maximization (EM). Στη συνέχεια, στο τέταρτο κεφάλαιο περιγράφονται οι πειραματικές διατάξεις που χρησιμοποιήθηκαν για την επαλήθευση της απόδοσης των αλγορίθμων και παρουσιάζονται τα αποτελέσματα των πειραμάτων που εκτελέστηκαν. Στο πέμπτο κεφάλαιο αναφέρονται κάποιες πιθανές επεκτάσεις και παρατίθενται τα συμπεράσματα από το σύνολο της παρούσας εργασίας, καθώς και η συνεισφορά της στο ερευνητικό πεδίο της ρομποτικής μάθησης.

Η παρούσα μεταπτυχιακή εργασία συνέβαλλε στη δημοσίευση των ακόλουθων εργασιών:

1. Nikos Vlassis, Kontes Georgios and Savas Piperidis, "Reinforcement Learning of Robot Control via Probabilistic Inference", 1st Hellenic Robotics Conference (HEROC), February 23-24, Athens, 2009

2. Nikos Vlassis, Marc Toussaint, Georgios Kontes and Savas Piperidis, "Learning Model - free Robot Control by a Monte Carlo EM Algorithm", Autonomous Robots, Special issue on robot learning (accepted)

Περιεχόμενα

Εισαγωγή **v**

Περιεχόμενα **vii**

1 Ενισχυτική Μάθηση **1**
1.1 Μαρκοβιανές Διεργασίες Απόφασης 1
1.2 Το βασικό μοντέλο της Ενισχυτικής Μάθησης 2
1.3 Ενισχυτική Μάθηση με Συναρτήσεις Αξιολόγησης 3
1.4 Αλγόριθμοι Πολιτικής Κλίσης 6

2 Ενισχυτική Μάθηση μέσω Πιθανοτικού Συμπερασμού **9**
2.1 Ο αλγόριθμος Expectation - Maximization 9
2.2 Ο αλγόριθμος Monte Carlo Expectation Maximization για Ενισχυτική Μάθηση 11
2.3 Ο αλγόριθμος Policy learning by Weighting Exploration with the Returns (PoWER) . . 14
2.4 Μέθοδοι Επιτάχυνσης των Αλγορίθμων Πιθανοτικού Συμπερασμού 16
2.4.1 Χρήση των καλύτερων δειγματικών τροχιών 16
2.4.2 Αναζήτηση Επί Γραμμής (Line Search) 17

3 Πειράματα **19**
3.1 Διδιάστατο Συνθετικό Πρόβλημα 19
3.2 Εξομοίωση αιώρησης ελικοπτέρου 25
3.3 RobBa (Robot Balancing) 28

4 Συμπεράσματα και Μελλοντικές Επεκτάσεις **33**
4.1 Συνεισφορά 33
4.2 Μελλοντικές Επεκτάσεις 34
4.3 Συμπεράσματα 35

Βιβλιογραφία **37**

Κεφάλαιο 1

Ενισχυτική Μάθηση

1.1 Μαρκοβιανές Διεργασίες Απόφασης

Για να ελέγξουμε μια διεργασία πρέπει να μπορούμε να παρατηρήσουμε κάθε στιγμή την κατάσταση στην οποία βρίσκεται, ποιες ενέργειες θα εκτελεστούν που θα την επηρεάσουν και τι αποτέλεσμα θα έχουν οι ενέργειες αυτές. Για παράδειγμα για να βρούμε την επόμενη θέση και ταχύτητα που θα έχει ένα αυτοκίνητο, πρέπει να γνωρίζουμε την προηγούμενη θέση του, πόσο έχουμε στρίψει το τιμόνι, πόση επιτάχυνση δίνουμε, ποια η ποιότητα του οδοστρώματος κ.τ.λ.

Η Υπόθεση Μάρκοβ (Markov Assumption) δηλώνει πως για να βρούμε την επόμενη κατάσταση του αυτοκινήτου δεν χρειάζεται να γνωρίζουμε τη θέση του και την ταχύτητά του όταν εκκινήσαμε το αυτοκίνητο, ούτε πόσο επιταχύναμε ή επιβραδύναμε το όχημα πριν μια ώρα, αλλά την κατάσταση του οχήματος την τρέχουσα χρονική στιγμή.

Ορισμός 1.1. *Υπόθεση Μάρκοβ (Markov Assumption): Η δυναμική ορισμένων συστημάτων μπορεί να περιγραφεί μόνο από την τρέχουσα κατάσταση του συστήματος και τη δράση που εφαρμόζεται σε αυτό και όχι από το πλήρες ιστορικό των καταστάσεων και των ενεργειών του συστήματος. Διαφορετικά, αν x_t είναι η τρέχουσα κατάσταση του συστήματος και u_t η τρέχουσα ενέργεια που εφαρμόζεται, τότε:* $Pr\{x_{t+1} = x' | x_t, u_t, x_{t-1}, u_{t-1}, \ldots, x_0, u_0\} = Pr\{x_{t+1} = x' | x_t, u_t\}$

Η Ενισχυτική Μάθηση χρησιμοποιείται για την επίλυση προβλημάτων, τα οποία μοντελοποιούνται με τη βοήθεια των Μαρκοβιανών Διεργασιών Απόφασης (ΜΔΑ ή Markov Decision Processes).

Ορισμός 1.2. *Μια ΜΔΑ (Bellman and Dreyfus, 1962; Howard, 1960; Puterman, 1994) είναι μια στοχαστική διεργασία διακριτού χρόνου, η οποία περιγράφεται από το διάνυσμα* (X, U, T, R, γ, D)*, όπου:*

- X *είναι ο χώρος καταστάσεων του προβλήματος, ο οποίος αντιπροσωπεύει τις διαφορετικές καταστάσεις στις οποίες μπορεί να βρεθεί η διεργασία κάθε χρονική στιγμή.*
- U *είναι ο χώρος των ενεργειών ή αποφάσεων της διεργασίας και αντιπροσωπεύει το σύνολο των διαφορετικών ενεργειών που μπορεί να εκτελέσει ο αποφασίζων κάθε χρονική στιγμή.*
- T *είναι το Μοντέλο Μετάβασης (Transition Model) της διεργασίας και περιγράφει την πιθανότητα αν η διεργασία βρίσκεται στην κατάσταση* x *τη χρονική στιγμή* t *και επιλεγεί η ενέργεια*

u, να βρεθεί η διεργασία σε μια νέα κατάσταση $\boldsymbol{x}'$ ή διαφορετικά $T(\boldsymbol{x}, u, \boldsymbol{x}') = T(\boldsymbol{x}'|\boldsymbol{x}, u)$. Γίνεται κατανοητό πως το μοντέλο μετάβασης ακολουθεί την υπόθεση Μάρκοβ.

- *R είναι η Συνάρτηση Ανταμοιβής (Reward Function) ή Συνάρτηση Κόστους (Cost Function) της διεργασίας, η οποία καθορίζει την ανταμοιβή που συλλέγει το σύστημα κάθε χρονική στιγμή. Η συνάρτηση ανταμοιβής είναι μια απεικόνιση των μεταβάσεων του συστήματος σε πραγματικούς αριθμούς, $R : X \times U \times X \to \mathbb{R}$. Και η συνάρτηση ανταμοιβής ακολουθεί την υπόθεση Μάρκοβ, δηλαδή η ανταμοιβή που λαμβάνει κάθε στιγμή το σύστημα ορίζεται ως $r_t = R(\boldsymbol{x}, u, \boldsymbol{x}')$ και δεν εξαρτάται από το ιστορικό του συστήματος, αλλά είναι πολύ συνηθισμένες και ανταμοιβές της μορφής $r_t = R(\boldsymbol{x}, u)$ ή και $r_t = R(\boldsymbol{x})$.*

- *$\gamma \in (0, 1]$ είναι ο παράγοντας έκπτωσης (discount factor) της διεργασίας, ο οποίος καθορίζει πόση επίδραση θα έχουν στη διεργασία ανταμοιβές που συλλέγονται μετά από t βήματα της διεργασίας.*

- *D είναι μια πιθανοτική κατανομή πάνω στο χώρο των καταστάσεων X, η οποία καθορίζει την αρχική κατάσταση της διεργασίας.*

1.2 Το βασικό μοντέλο της Ενισχυτικής Μάθησης

Στην Ενισχυτική Μάθηση (Bertsekas and Tsitsiklis, 1996; Kaelbling et al., 1996; Sutton and Barto, 1998) θεωρούμε πως το πρόβλημα μπορεί να διαχωριστεί στο σύστημα (ή περιβάλλον) και στον αποφασίζοντα (decision maker). Θεωρούμε πως σε κάθε χρονική στιγμή το σύστημα βρίσκεται σε μια κατάσταση $\boldsymbol{x} \in X$ και ο αποφασίζων εφαρμόζει πάνω του μια ενέργεια $u \in U$, η οποία έχει σαν συνέπεια τη μεταφορά του συστήματος σε μια επόμενη κατάσταση $\boldsymbol{x}' \in X$, και την συλλογή μιας ανταμοιβής $r \in \mathbb{R}$. Στην παρούσα εργασία, όπου ασχολούμαστε με πραγματικά ρομποτικά συστήματα οι καταστάσεις και οι ενέργειες θα ανήκουν σε συνεχείς χώρους, δηλαδή $X \subseteq \mathbb{R}^N$ και $U \subseteq \mathbb{R}^M$.[1]

Για να μπορέσουμε να ελέγξουμε το σύστημα πρέπει σε κάθε χρονική στιγμή να επιλέγουμε την κατάλληλη ενέργεια για την εργασία που θέλουμε να εκτελέσουμε. Η συνάρτηση η οποία επιλέγει κάθε στιγμή μια ενέργεια, ανάλογα με την κατάσταση του συστήματος ονομάζεται πολιτική.

Ορισμός 1.3. *Πολιτική (Policy) ή ελεγκτής (controller) ονομάζεται η αντιστοίχιση $\pi(\boldsymbol{s}) : S \to A$ από καταστάσεις σε ενέργειες. Για την παρούσα εργασία θα θεωρούμε ότι η πολιτική είναι της μορφής $u_t \sim \pi_{\boldsymbol{\theta}}(u_t|\boldsymbol{x}_t) = p(u_t|\boldsymbol{x}_t, \boldsymbol{\theta})$, όπου το διάνυσμα $\boldsymbol{\theta} \in \mathbb{R}^L$ δηλώνει τις L παραμέτρους της πολιτικής $\pi_{\boldsymbol{\theta}}$. Η πολιτική εδώ είναι στοχαστική, χρησιμεύοντας στην εξερεύνηση νέων καταστάσεων του συστήματος.*

Συνοψίζοντας όλα τα προηγούμενα έχουμε:

Ορισμός 1.4. *Ένα Μαρκοβιανό Σύστημα ορίζεται από τις εξισώσεις:*

[1] Στην παρούσα εργασία θεωρούμε $M = 1$

$$\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0), \tag{1.1}$$

$$\boldsymbol{x}_{t+1} \sim p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t), \tag{1.2}$$

$$r_{t+1} = R(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{x}_{t+1}), \tag{1.3}$$

$$\boldsymbol{u}_t \sim \pi_{\boldsymbol{\theta}}(\boldsymbol{u}_t|\boldsymbol{x}_t) = p(\boldsymbol{u}_t|\boldsymbol{x}_t, \boldsymbol{\theta}), \tag{1.4}$$

με καταστάσεις $\boldsymbol{x}_t \in X \subseteq \mathbb{R}^N$, ενέργειες $u_t \in U \subseteq \mathbb{R}^M$ για όλες τις χρονικές στιγμές $t \in \mathbb{N}$ και παραμέτρους $\boldsymbol{\theta} \in \mathbb{R}^L$.

Εκτός από τις παραπάνω εξισώσεις πρέπει να γνωρίζουμε για πόσο χρονικό διάστημα θα λειτουργεί το σύστημά μας, πρέπει να γνωρίζουμε δηλαδή αν η διεργασία είναι άπειρη ή διακόπτεται μετά από κάποια βήματα. Ο ορίζοντας μιας διεργασίας είναι ο "χρόνος ζωής" της διεργασίας. Ο ορίζοντας μπορεί να είναι άπειρος (η διεργασία εκτελείται για πάντα) ή πεπερασμένος (η διεργασία τερματίζει με πιθανότητα 1 μετά από κάποιο αριθμό βημάτων, ο οποίος δεν είναι γνωστός εκ των προτέρων) ή δεδομένος (η διεργασία εκτελείται για συγκεκριμένο αριθμό βημάτων).

Μια πλήρης εκτέλεση των εξισώσεων 1.1 - 1.4 έως τον ορίζοντα H ονομάζεται επεισόδιο ή τροχιά (rollout ή trajectory ή episode) και αποτελείται από μια ακολουθία από καταστάσεις, ενέργειες και ανταμοιβές:

$$\boldsymbol{x}_0 \xrightarrow[r_0]{u_0} \boldsymbol{x}_1 \xrightarrow[r_1]{u_1} \boldsymbol{x}_2 \xrightarrow[r_2]{u_2} \boldsymbol{x}_3 \xrightarrow[r_3]{u_3} \boldsymbol{x}_4 \xrightarrow[r_4]{u_4} \quad \dots \quad \boldsymbol{x}_{H-1} \xrightarrow[r_{H-1}]{u_{H-1}} \boldsymbol{x}_H \tag{1.5}$$

Ας υποθέσουμε τώρα πως έχουμε ένα σύστημα, το οποίο αλληλεπιδρά με το περιβάλλον και δημιουργεί επεισόδια και πως έχουμε επιλέξει μια παραμετροποιημένη πολιτική, η οποία παράγει τις ενέργειες του συστήματος. Σκοπός μας είναι να μεγιστοποιήσουμε τις ανταμοιβές που συλλέγει το σύστημα για όλες τις πιθανές τροχιές που μπορεί να παραχθούν. Το μέτρο που χρησιμοποιείται είναι η αναμενόμενη ανταμοιβή (expected total discounted reward ή expected cumulative reward ή value of policy ή expected return). Πλέον μπορούμε να ορίσουμε το πλαίσιο της Ενισχυτικής Μάθησης:

Ορισμός 1.5. *Σκοπός της Ενισχυτικής Μάθησης είναι να βρει την πολιτική $\pi^*_{\boldsymbol{\theta}}$ με παραμέτρους $\boldsymbol{\theta}^* \in \mathbb{R}^L$, η οποία είναι βέλτιστη ως προς μια αντικειμενική συνάρτηση (evaluation function) $J(\boldsymbol{\theta})$, χρησιμοποιώντας μόνο δειγματική εμπειρία από το σύστημα. Η πιο συνηθισμένη αντικειμενική συνάρτηση είναι η αναμενόμενη ανταμοιβή, δηλαδή θα έχουμε:*

$$J(\boldsymbol{\theta}) = \mathbb{E}\left\{\sum_{t=0}^{H} \gamma^t r_t; \boldsymbol{\theta}\right\}, \tag{1.6}$$

όπου η αναμενόμενη τιμή $\mathbb{E}\{\cdot\}$ αφορά όλες τις πιθανές τροχιές που μπορούν να προκύψουν ξεκινώντας από την κατάσταση $\boldsymbol{x}_0$ και ακολουθώντας την $\pi_{\boldsymbol{\theta}}$.

1.3 Ενισχυτική Μάθηση με Συναρτήσεις Αξιολόγησης

Πολλοί αλγόριθμοι Ενισχυτικής Μάθησης στηρίζονται στην εκτίμηση κάποιων συναρτήσεων αξιολόγησης. Για μια Μαρκοβιανή Διεργασία Απόφασης η συνάρτηση αξιολόγησης κατάστασης (state

value function) V αποδίδει μια τιμή σε μια κατάσταση x του συστήματος. Η τιμή $V^{\pi}(x)$ μιας κατάστασης x με πολιτική π είναι η αναμενόμενη ανταμοιβή που συλλέγει το σύστημα όταν ξεκινά από την κατάσταση x και ακολουθεί πολιτική π:

$$V^{\pi}(x) = \mathbb{E}_{\pi}\left\{\sum_{t=0}^{H}\gamma^{t}r_{t}|x_{0}=x\right\} \tag{1.7}$$

Παρόμοια, η συνάρτηση αξιολόγησης κατάστασης - ενέργειας (state - action value function) αποδίδει μια τιμή σε κάθε ζεύγος (x, u) καταστάσεων και ενεργειών. Η τιμή $Q^{\pi}(x, u)$ της επιλογής να εκτελεστεί η ενέργεια u όταν το σύστημα βρίσκεται σε κατάσταση x και ακολουθεί πολιτική π είναι η αναμενόμενη ανταμοιβή που συλλέγεται όταν το σύστημα βρίσκεται σε κατάσταση x, εκτελείται η ενέργεια u στο πρώτο βήμα και ακολουθείται στη συνέχεια η πολιτική π:

$$Q^{\pi}(x,u) = \mathbb{E}_{\pi}\left\{\sum_{t=0}^{H}\gamma^{t}r_{t}|x_{0}=x,u_{0}=u\right\} \tag{1.8}$$

Η συνάρτηση αξιολόγησης κατάστασης και η συνάρτηση αξιολόγησης κατάστασης - ενέργειας συνδέονται με την ακόλουθη σχέση:

$$Q^{\pi}(x,u) = r(x,u) + \gamma\sum_{x'\in X}T(x,u,x')V^{\pi}(x'), \tag{1.9}$$

Δεδομένης μιας πολιτικής π, η άπληστη πολιτική (greedy policy) π' πάνω στην π (over π) είναι μια αιτιοκρατική πολιτική η οποία μπορεί να υπολογιστεί απ' ευθείας από την Q^{π} ή τη V^{π}. Ειδικότερα, είναι η πολιτική η οποία μεγιστοποιεί την Q^{π} σε κάθε κατάσταση:

$$\pi'(x) = \arg\max_{u\in U}Q^{\pi}(x,u) \tag{1.10}$$

και ισχύει

$$\forall x\in X,\quad Q^{\pi'}(x,\pi'(x)) \geq Q^{\pi}(x,\pi(x)) \tag{1.11}$$

Για κάθε πολιτική π η γραμμική εξίσωση Bellman συνδέει τις τιμές Q^{π} μεταξύ ζευγών καταστάσεων και ενεργειών. Για αιτιοκρατικές πολιτικές ισχύει:

$$Q^{\pi}(x,u) = r(x,u) + \gamma\sum_{x'\in X}T(x,u,x')Q^{\pi}(x',\pi(x')) \tag{1.12}$$

Οι τιμές για της Q^{π} για όλα τα ζεύγη καταστάσεων - ενεργειών μπορούν να βρεθούν αν λυθεί το γραμμικό σύστημα διαστάσεων $|X||U| \times |X||U|$, που προκύπτει από τις εξισώσεις Bellman για όλα τα ζεύγη καταστάσεων - ενεργειών.

Αντίθετα, η εξίσωση βελτιστότητας Bellman (Bellman optimality equation), η οποία συνδέει τις τιμές της βέλτιστης συνάρτησης Q^{*} της βέλτιστης πολιτικής π^{*} είναι μη γραμμική και καταλήγει σε ένα μη γραμμικό σύστημα εξισώσεων:

$$Q^{*}(x,u) = r(x,u) + \gamma\sum_{x'\in X}T(x,u,x')\max_{u'\in U}Q^{*}(x',u') \tag{1.13}$$

Η εξίσωση δείχνει πως οι επιλογές των ενεργειών της βέλτιστης πολιτικής μεγιστοποιούν την αναμενόμενη ανταμοιβή.

Η επίλυση μιας ΜΔΑ είναι η εύρεση της βέλτιστης πολιτικής π^* για την ΜΔΑ. Οι κλασικές μέθοδοι για την εύρεση της π^* είναι:

- Value Iteration (Bellman and Dreyfus, 1962; Bertsekas, 1987; Bertsekas and Tsitsiklis, 1996): Ξεκινά με ένα αρχικό διάνυσμα Q και εφαρμόζοντας συνεχώς την εξίσωση βελτιστότητας Bellman προσεγγίζει την τιμή της βέλτιστης Q^*. Στην συνέχεια υπολογίζει την π^*, η οποία είναι η άπληστη πολιτική πάνω στην Q^*.

- Policy Iteration (Howard, 1960; Bertsekas and Tsitsiklis, 1996): Ξεκινά με μια αρχική πολιτική π, υπολογίζει τις τιμές τις Q^π επιλύοντας τις γραμμικές εξισώσεις Bellman και εξάγει την νέα πολιτική π', η οποία είναι η άπληστη πολιτική πάνω στην Q^π. Τα δύο αυτά βήματα επαναλαμβάνονται μέχρι $\pi' = \pi^*$.

Οι κλασικές αυτές μέθοδοι, οι οποίες στηρίζονται στις εξισώσεις Bellman απαιτούν γνώση του συστήματος, αφού χρησιμοποιούν το μοντέλο μετάβασης T. Για να ξεπεραστεί το εμπόδιο αυτό έχουν αναπτυχθεί πολλοί αλγόριθμοι (Q-Learning, Fitted-Q, LSPI κ.ά) (Watkins, 1989; Ernst et al., 2006; Lagoudakis and Parr, 2003), οι οποίοι επιχειρούν να μάθουν τις τιμές Q από δείγματα που συλλέγονται κατά την εκτέλεση της ΜΔΑ.

Μια δυσκολία για τη μεταφορά των συγκεκριμένων αλγορίθμων Ενισχυτικής Μάθησης σε πραγματικά προβλήματα είναι η διακριτοποίηση που απαιτείται στο χώρο καταστάσεων και ενεργειών. Η χρήση της συνάρτησης Q (και V) προϋποθέτει την κατάτμηση τν X και U, αφού περιέχει ζεύγη διακριτών καταστάσεων και ενεργειών. Για να καμφθεί το μειονέκτημα της διακριτοποίησης του χώρου καταστάσεων, διάφοροι αλγόριθμοι (π.χ. LSPI (Lagoudakis and Parr, 2003)) χρησιμοποιούν συναρτήσεις βάσεις για να προσεγγίσουν τις τιμές της Q: $Q^\pi(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{\phi}(x, u)^T \boldsymbol{w}$, όπου $\boldsymbol{w}$ είναι οι παράμετροι της προσέγγισης και $\boldsymbol{\phi}(x, u)$ οι συναρτήσεις βάσης. Ακόμη, όμως, και να παρέχονται οι ιδανικές συναρτήσεις βάσης, όσες τροχιές και να συλλέξουμε από το σύστημα δεν αρκούν για να καλυφθεί ο εκθετικά μεγάλος χώρος καταστάσεων που απαιτείται για τα πραγματικά ρομποτικά συστήματα. Επιπλέον, πάντα παραμένει το πρόβλημα των διακριτών ενεργειών και των ανανεώσεων των πολιτικών με τη χρήση του τελεστή απληστίας. Όλα τα παραπάνω συντελούν στην παραγωγή πολιτικών που αποδεικνύονται εξαιρετικά ασταθείς σε πραγματικά ρομποτικά προβλήματα.

Κάποιοι άλλοι αλγόριθμοι (Gaskett et al., 1999; Santamaria et al., 1997; Lazaric et al., 2007; Pazis and Lagoudakis, 2009) προσπαθούν να αντιμετωπίσουν το πρόβλημα της διακριτοποίησης των ενεργειών είτε χρησιμοποιώντας νευρωνικά δίκτυα ή γραμμική παλινδρόμηση είτε προσπαθώντας να βρουν βέλτιστες διακριτοποιήσεις. Ακόμη όμως και με βέλτιστες διακριτοποιήσεις του χώρου των ενεργειών, οι ενέργειες δεν παύουν να είναι πεπερασμένες. Το γεγονός αυτό μειώνει την ακρίβεια και την ποιότητα του ελεγκτή, καθιστώντας τον ακατάλληλο για πραγματικά ρομποτικά συστήματα που απαιτούν αυξημένη ακρίβεια (π.χ. ρομπότ συναρμολόγησης ή χειρουργικά ρομπότ). Γίνεται λοιπόν φανερή, η ανάγκη ανάπτυξης αλγορίθμων που να είναι σε θέση να αντιμετωπίσουν προβλήματα ρομποτικού ελέγχου πραγματικών συστημάτων.

1.4 Αλγόριθμοι Πολιτικής Κλίσης

Μια άλλη μεγάλη οικογένεια αλγορίθμων στον τομέα της Ενισχυτικής Μάθησης αποτελούν οι αλγόριθμοι αναζήτησης πολιτικής (policy search). Οι αλγόριθμοι αυτοί αναζητούν τη βέλτιστη πολιτική εργαζόμενοι απ' ευθείας στο χώρο των πολιτικών. Οι πιο απλοί αλγόριθμοι αναζήτησης πολιτικής είναι οι αλγόριθμοι πολιτικής κλίσης (policy gradient) (Williams, 1992; Sutton et al., 2000; Peters and Schaal, 2006; Lawrence et al., 2003). Ένας αλγόριθμος πολιτικής κλίσης υπολογίζει και ακολουθεί σε κάθε βήμα το στοχαστικό διάνυσμα κλίσης (gradient) της αναμενόμενης ανταμοιβής $J(\boldsymbol{\theta})$ της πολιτικής, μέχρι να συγκλίνει σε κάποιο (τοπικό) μέγιστο, δηλαδή:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha_k \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_k} + \alpha_k \boldsymbol{\varepsilon}_k, \tag{1.14}$$

όπου $\alpha_k \in \mathbb{R}^+$ είναι μια ακολουθία από ρυθμούς μάθησης και ο όρος $\boldsymbol{\varepsilon}_k \in \mathbb{R}^L$ δηλώνει το λάθος στην εκτίμηση του διανύσματος κλίσης σε σχέση με το πραγματικό. Οι αλγόριθμοι πολιτικής κλίσης συγκλίνουν εγγυημένα σε τοπικό ελάχιστο αν ισχύουν τα ακόλουθα (Duflo and Wilson, 1997):

1. Για τους ρυθμούς μάθησης να ισχύει: $\sum_{k=0}^{k=\infty} \alpha_k = \infty$ και $\sum_{k=0}^{k=\infty} \alpha_k^2 < \infty$
2. Η γωνία ανάμεσα στο πραγματικό διάνυσμα κλίσης $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_k}$ και στο εκτιμώμενο $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_k} + \alpha_k \boldsymbol{\varepsilon}_k$ να είναι μικρότερη από 90°, δηλαδή $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_k}^T (\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_k} + \mathbb{E}\{\boldsymbol{\varepsilon}_k\}) > \mathbf{0}$
3. Τα λάθη στην εκτίμηση του διανύσματος κλίσης να είναι φραγμένα, δηλαδή $\boldsymbol{\varepsilon}_k^T \boldsymbol{\varepsilon}_k < \infty$

Το σημαντικότερο πρόβλημα στις μεθόδους πολιτικής κλίσης είναι η εκτίμηση του διανύσματος κλίσης μόνο από δεδομένα (data) του συστήματος και χωρίς γνώση του μοντέλου του περιβάλλοντος και της διεργασίας. Υπάρχουν δύο κύριες οικογένειες μεθόδων για την εκτίμηση του διανύσματος κλίσης για τη βελτιστοποίηση στοχαστικών συστημάτων (Glynn, 1987), οι μέθοδοι των διακριτών διαφορών (finite-difference) και η μέθοδοι πιθανοφάνειας (likelihood ratio). Οι μέθοδοι διακριτών διαφορών επιφέρουν μικρές αλλαγές $\Delta\boldsymbol{\theta}$ στις παραμέτρους της πολιτικής και στη συνέχεια εκτελώντας διάφορες τροχιές στο σύστημα με τις νέες παραμέτρους, υπολογίζουν μια εκτίμηση για την ποιότητα των διαφοροποιημένων παραμέτρων:

$$\Delta\hat{J} \approx J(\boldsymbol{\theta}_k + \Delta\boldsymbol{\theta}) - J_{ref}{}^2 \tag{1.15}$$

Στη συνέχεια για τον υπολογισμό του διανύσματος κλίσης g_{FD} αρκεί η επίλυση ενός προβλήματος γραμμικής παλινδρόμησης:

$$\boldsymbol{g}_{FD} = (\Delta\boldsymbol{\theta}^T \Delta\boldsymbol{\theta})^{-1} \Delta\boldsymbol{\theta}^T \Delta\hat{J} \tag{1.16}$$

Οι μέθοδοι διακριτών διαφορών, αν και είναι πολύ απλές στην κατανόηση και την υλοποίησή τους και αποδίδουν ικανοποιητικά σε ντετερμινιστικά προβλήματα (Spall, 2005; Fidelman and Stone, 2004; Kohl and Stone, 2004), υστερούν σε απόδοση σε σχέση με τις μεθόδους πιθανοφάνειας (Glynn, 1987). Αν υποθέσουμε ότι το σύστημα παράγει τροχιές $\xi \sim p_{\boldsymbol{\theta}}(\xi) = p(\xi|\boldsymbol{\theta})$ με $r(\xi) = \sum_{t=0}^{H} \alpha_t r_t$, τότε μπορούμε να υπολογίσουμε το διάνυσμα κλίσης χρησιμοποιώντας το REINFORCE trick (Williams, 1992):

[2] Ως J_{ref} χρησιμοποιείται συνήθως το $J(\boldsymbol{\theta}_k)$ ή το $J(\boldsymbol{\theta}_k - \Delta\boldsymbol{\theta})$

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_{\Xi} \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(\xi) r(\xi) = \mathbb{E}\{\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\xi) r(\xi)\} \tag{1.17}$$

Στη συνέχεια έχουμε:

$$p_{\boldsymbol{\theta}}(\xi) = p(\boldsymbol{x}_0) \prod_{t=0}^{H} p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t) \pi_{\boldsymbol{\theta}}(\boldsymbol{u}_t|\boldsymbol{x}_t) \Longrightarrow \nabla_{\boldsymbol{\theta}} log p_{\boldsymbol{\theta}}(\xi) = \sum_{t=0}^{H} \nabla_{\boldsymbol{\theta}} log \pi_{\boldsymbol{\theta}}(\boldsymbol{u}_t|\boldsymbol{x}_t), \tag{1.18}$$

οπότε ο υπολογισμός του διανύσματος κλίσης είναι πια συνάρτηση μόνο των ανταμοιβών που λαμβάνει το σύστημα σε κάθε τροχιά $r(\xi)$ και της πολιτικής που ακολουθεί $\pi_{\boldsymbol{\theta}}$. Επειδή η εκτίμηση που προκύπτει για το διάνυσμα κλίσης έχει μεγάλη διασπορά, μπορούμε να αφαιρέσουμε μια σταθερή ποσότητα (baseline) $\boldsymbol{b}$, οπότε τελικά το διάνυσμα κλίσης θα είναι:

$$\boldsymbol{g}_{RF} = \left\langle \left(\sum_{t=0}^{H} \nabla_{\boldsymbol{\theta}} log \pi_{\boldsymbol{\theta}} (\boldsymbol{u}_t|\boldsymbol{x}_t) \right) \left(\sum_{l=0}^{H} a_l r_l - \boldsymbol{b} \right) \right\rangle_{\xi}, \tag{1.19}$$

όπου $\langle \cdot \rangle_{\xi}$ δηλώνει το δειγματικό μέσο ως προς το πλήθος των τροχιών.

Από την άλλη πλευρά οι μελλοντικές ενέργειες του συστήματος δεν εξαρτώνται από τις προηγούμενες ανταμοιβές, οπότε το διάνυσμα κλίσης υπολογίζεται σύμφωνα με τον αλγόριθμο G(PO)MDP (Sutton et al., 2000):

$$\boldsymbol{g}_{G(PO)MDP} = \left\langle \sum_{t=0}^{H} \left(\sum_{k=0}^{t} \nabla_{\boldsymbol{\theta}} log \pi_{\boldsymbol{\theta}}(\boldsymbol{u}_k|\boldsymbol{x}_k) \right) (\alpha_t r_t - \boldsymbol{b}_t) \right\rangle_{\xi} \tag{1.20}$$

Ακόμη και αν υπάρχει η βέλτιστη baseline η σύγκλιση των αλγορίθμων τύπου REINFORCE και G(PO)MDP είναι αρκετά αργή. Ο ταχύτερος αλγόριθμος πολιτικών κλίσεων μέχρι στιγμής είναι ο (episodic) Natural Actor Critic (Peters and Schaal, 2008), ο οποίος συνδυάζει (στη βασική του μορφή) τα στοιχεία των προηγούμενων αλγορίθμων με μια γραμμική παλινδρόμηση. Έτσι αν έχουμε τις τροχιές $\xi_1, \xi_2, \ldots, \xi_s$ και $\boldsymbol{\theta} \in \mathbb{R}^L$, τότε θα έχουμε:

αν

$$\boldsymbol{X} = \begin{bmatrix} \sum_{t=0}^{H} \nabla log \pi_{\boldsymbol{\theta}_1}(\boldsymbol{u}_t^1|\boldsymbol{x}_t^1) & \sum_{t=0}^{H} \nabla log \pi_{\boldsymbol{\theta}_2}(\boldsymbol{u}_t^1|\boldsymbol{x}_t^1) & \ldots & \sum_{t=0}^{H} \nabla log \pi_{\boldsymbol{\theta}_L}(\boldsymbol{u}_t^1|\boldsymbol{x}_t^1) & 1 \\ \sum_{t=0}^{H} \nabla log \pi_{\boldsymbol{\theta}_1}(\boldsymbol{u}_t^2|\boldsymbol{x}_t^2) & \sum_{t=0}^{H} \nabla log \pi_{\boldsymbol{\theta}_2}(\boldsymbol{u}_t^2|\boldsymbol{x}_t^2) & \ldots & \sum_{t=0}^{H} \nabla log \pi_{\boldsymbol{\theta}_L}(\boldsymbol{u}_t^2|\boldsymbol{x}_t^2) & 1 \\ \vdots & \vdots & & \vdots & \\ \sum_{t=0}^{H} \nabla log \pi_{\boldsymbol{\theta}_1}(\boldsymbol{u}_t^s|\boldsymbol{x}_t^s) & \sum_{t=0}^{H} \nabla log \pi_{\boldsymbol{\theta}_2}(\boldsymbol{u}_t^s|\boldsymbol{x}_t^s) & \ldots & \sum_{t=0}^{H} \nabla log \pi_{\boldsymbol{\theta}_L}(\boldsymbol{u}_t^s|\boldsymbol{x}_t^s) & 1 \end{bmatrix} \tag{1.21}$$

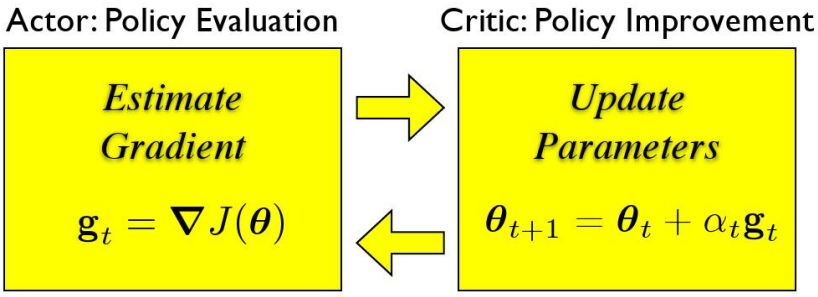
και

$$\boldsymbol{Y} = \left[\begin{array}{cccc} \sum_{t=0}^{H} r_t^1 & \sum_{t=0}^{H} r_t^2 & \cdots & \sum_{t=0}^{H} r_t^s \end{array} \right]^T, \tag{1.22}$$

τότε η εκτίμηση του διανύσματος κλίσης θα δίνεται από τη σχέση:

$$\left[\begin{array}{c} \boldsymbol{g} \\ J \end{array} \right] = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{Y} \tag{1.23}$$

Οι αλγόριθμοι της παραγράφου αυτής ονομάζονται και Actor - Critic αλγόριθμοι (Barto et al., 1988), αφού όπως είδαμε αποτελούνται από δύο στάδια, τα οποία επαναλαμβάνονται διαδοχικά, έως ότου επιτευχθεί σύγκλιση στην βέλτιστη πολιτική. Στο πρώτο στάδιο (Actor) εκτελούνται ένας αριθμός από τροχιές στο σύστημα και με τα δεδομένα που συλλέγονται κατά την εκτέλεσή τους υπολογίζεται μια εκτίμηση για το διάνυσμα κλίσης. Στη συνέχεια, στο δεύτερο στάδιο (Critic) υπολογίζονται οι νέες παράμετροι της πολιτικής, με βάση την εκτίμηση για το διάνυσμα κλίσης. Η διαδικασία φαίνεται σχηματικά στην εικόνα 1.1.



Σχήμα 1.1: Δομή των Actor - Critic αλγορίθμων

Στα θετικά των αλγορίθμων αυτού του κεφαλαίου συγκαταλέγονται η χρήση αμιγώς συνεχών χώρων καταστάσεων και ενεργειών, καθώς και η δυνατότητα χρήσης ελεγκτών που έχουν φυσική σημασία σε σχέση με το σύστημα και όχι μιας υπερπληθούς σειράς από συναρτήσεις βάσης.

Από την άλλη πλευρά, για να είναι αποδοτικοί, απαιτείται τέλεια ρύθμιση όλων των ρυθμών μάθησης. Αν οι ρυθμοί μάθησης μειώνονται πολύ γρήγορα ή είναι πολύ μεγάλοι, τότε δεν θα επιτευχθεί σύγκλιση. Επιπλέον, είναι ευαίσθητοι σε τοπικά βέλτιστα, οπότε απαιτείται ένα αρκετά καλό σετ αρχικών παραμέτρων.

Κεφάλαιο 2

Ενισχυτική Μάθηση μέσω Πιθανοτικού Συμπερασμού

Όπως έγινε αντιληπτό από την προηγούμενη ενότητα το μεγαλύτερο πρόβλημα των αλγορίθμων πολιτικής κλίσης είναι η εύρεση των τιμών των ρυθμών μάθησης. Το ιδανικό θα ήταν η ύπαρξη ενός αλγορίθμου, ο οποίος δεν θα υπολόγιζε απλά την κατεύθυνση προς την οποία πρέπει να κινηθεί ο αλγόριθμος στο χώρο των παραμέτρων, αλλά θα παρείχε στην έξοδο ένα νέο διάνυσμα παραμέτρων απ' ευθείας.

Προς αυτή την κατεύθυνση υπάρχουν κάποιες πρόσφατες ερευνητικές εργασίες στον τομέα της Ενισχυτικής Μάθησης, οι οποίες στηρίζονται στη χρήση του αλγορίθμου Expectation - Maximization (EM) και προσπαθούν να επιλύσουν το πρόβλημα της Ενισχυτικής Μάθησης μέσω πιθανοτικού συμπερασμού (probabilistic inference).

2.1 Ο αλγόριθμος Expectation - Maximization

Ο αλγόριθμος EM (Dempster et al., 1977; Neal and Hinton, 1998) είναι μια τεχνική για την εύρεση λύσεων μέγιστης πιθανοφάνειας (maximum likelihood) πιθανοτικών μοντέλων με λανθάνουσες (latent) μεταβλητές. Έστω πιθανοτικό μοντέλο με παρατηρήσιμες μεταβλητές $\boldsymbol{X}$ και λανθάνουσες μεταβλητές $\boldsymbol{Z}$ και έστω ότι η συνδυασμένη κατανομή (joint distribution) των $\boldsymbol{X}$ και $\boldsymbol{Z}$ εξαρτάται και από κάποιες παραμέτρους $\boldsymbol{\theta}$. Σκοπός ας είναι να μεγιστοποήσουμε ως προς $\boldsymbol{\theta}$ τη συνάρτηση πιθανοφάνειας:

$$p(\boldsymbol{X}|\boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta}) \tag{2.1}$$

Πρώτο μας βήμα είναι να εισάγουμε μια κατανομή $q(\boldsymbol{Z})$ ορισμένη πάνω στις λανθάνουσες μεταβλητές. Τότε παρατηρούμε πως για κάθε επιλογή της $q(\boldsymbol{Z})$ ισχύει η ακόλουθη σχέση:

$$\ln p(\boldsymbol{X}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL(q||p), \tag{2.2}$$

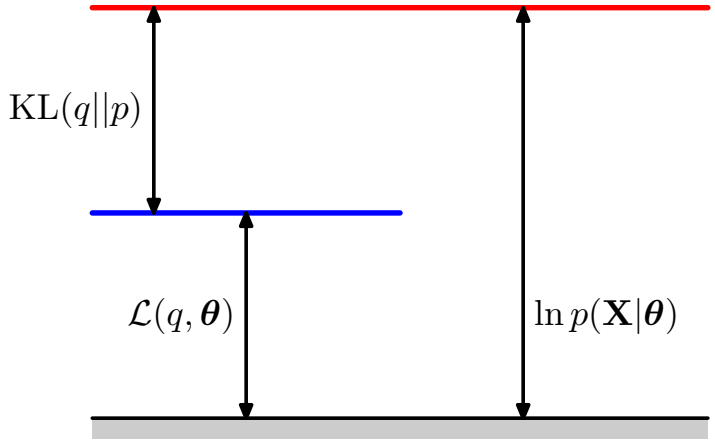
όπου έχουμε ορίσει:

$$\mathcal{L}(q,\boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \ln \left\{ \frac{p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})}{q(\boldsymbol{Z})} \right\} \tag{2.3}$$

και

$$KL(q||p) = -\sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \ln \left\{ \frac{p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta})}{q(\boldsymbol{Z})} \right\} \tag{2.4}$$

Χρησιμοποιούμε το συμβολισμό KL για την απόσταση Kullback - Leibler ανάμεσα στην $q(\boldsymbol{Z})$ και την εκ την υστέρων κατανομή (posterior distribution) $p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta})$. Για την Kullback - Leibler ισχύει πάντα ότι $KL > 0$ και κατ' εξαίρεσιν ότι $KL = 0$, αν και μόνο αν $q(\boldsymbol{Z}) = p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta})$. Οπότε, όπως φαίνεται και από τη σχέση 2.2, $\mathcal{L}(q,\boldsymbol{\theta}) \leq \ln p(\boldsymbol{X}|\boldsymbol{\theta})$ και η $\mathcal{L}(q,\boldsymbol{\theta})$ είναι ένα κάτω όριο της $\ln p(\boldsymbol{X}|\boldsymbol{\theta})$. Η σχέση 2.2 φαίνεται σχηματικά στο σχήμα 2.1[1].



Σχήμα 2.1: Σχηματοποίηση της σχέσης 2.2

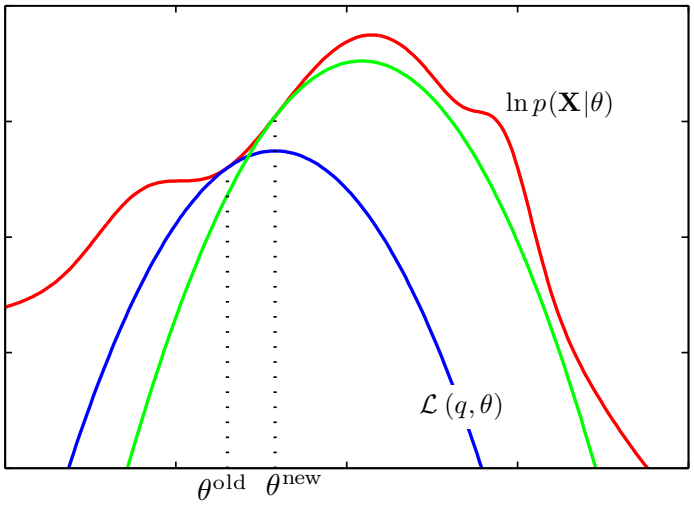
Ο αλγόριθμος ΕΜ είναι ένας επαναλαμβανόμενος αλγόριθμος δύο βημάτων (Ε και Μ), για την εύρεση λύσεων μεγιστοποίησης πιθανοφάνειας. Έστω ότι έχουμε ένα διάνυσμα παραμέτρων $\boldsymbol{\theta}_{old}$. Στο Ε βήμα του αλγορίθμου το κάτω όριο $\mathcal{L}(q,\boldsymbol{\theta}_{old})$ μεγιστοποιείται ως προς την $q(\boldsymbol{Z})$ και το διάνυσμα $\boldsymbol{\theta}_{old}$ διατηρείται σταθερό. Η μέγιστη τιμή της $\mathcal{L}(q,\boldsymbol{\theta}_{old})$ επιτυγχάνεται όταν η $KL(q||p)$ μηδενίζεται, όταν δηλαδή $q(\boldsymbol{Z}) = p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}_{old})$, κάτι που φαίνεται ξεκάθαρα και από την εικόνα 2.1. Στο Μ βήμα του αλγορίθμου, η $q(\boldsymbol{Z})$ διατηρείται σταθερή και το κάτω όριο $\mathcal{L}(q,\boldsymbol{\theta})$ μεγιστοποιείται ως προς $\boldsymbol{\theta}$, δίνοντας ένα διάνυσμα $\boldsymbol{\theta}_{new}$.

Η λειτουργία του αλγορίθμου ΕΜ γίνεται ευκολότερα αντιληπτή στο χώρο των παραμέτρων, όπως φαίνεται και στην εικόνα 2.2[2]. Η κόκκινη γραμμή είναι η λογαριθμική πιθανοφάνεια (log likelihood) που θέλουμε μεγιστοποιήσουμε. Αρχίζουμε τη διαδικασία με ένα αρχικό $\boldsymbol{\theta}_{old}$ και στο Ε βήμα του αλγορίθμου υπολογίζουμε την εκ των υστέρων κατανομή πάνω στις λανθάνουσες μεταβλητές, οπότε το κάτω όριο $\mathcal{L}(q,\boldsymbol{\theta})$ ανεβαίνει και ακουμπά τη log likelihood στο $\boldsymbol{\theta}_{old}$, όπως φαίνεται από τη μπλε γραμμή. Το όριο είναι κοίλη (concave) συνάρτηση με μοναδικό μέγιστο. Στο βήμα Μ βρίσκουμε το

[1] Η εικόνα είναι από το βιβλίο Pattern Recognition and Machine Learning του C.M.Bishop

[2] Η εικόνα είναι από το βιβλίο Pattern Recognition and Machine Learning του C.M.Bishop

μέγιστο του ορίου $\boldsymbol{\theta}_{new}$, το οποίο θα αντιστοιχεί σε μεγαλύτερη τιμή της log likelihood από το $\boldsymbol{\theta}_{old}$. Η διαδικασία επαναλαμβάνεται και το νέο βήμα Ε φτιάχνει ένα νέο κάτω όριο, στο $\boldsymbol{\theta}_{new}$ αυτή τη φορά, το οποίο φαίνεται με πράσινη γραμμή.



Σχήμα 2.2: Ο αλγόριθμος ΕΜ στο χώρο των παραμέτρων

2.2 Ο αλγόριθμος Monte Carlo Expectation Maximization για Ενισχυτική Μάθηση

Στην ενότητα αυτή, παρουσιάζεται η εφαρμογή του αλγορίθμου ΕΜ στην ανάπτυξη ρομποτικών ελεγκτών μέσω ενισχυτικής μάθησης. Στην προσέγγιση αυτή, ο ορίζοντας της Μαρκοβιανής Διεργασίας Απόφασης θεωρείται διακριτή τυχαία μεταβλητή με γεωμετρική κατανομή $p(T) = (1-\gamma)\gamma^T, \quad T = 0, 1, \ldots, \infty$. Η κεντρική ιδέα του αλγορίθμου είναι να θεωρήσουμε τις ανταμοιβές σαν πιθανότητες κάποιων φανταστικών γεγονότων (για το λόγο αυτό απαιτείται $r \in [0, 1]$) (Cooper, 1988). Συγκεκριμένα, η ανταμοιβή r_T που συλλέχθηκε σε κάποιο βήμα T της άπειρου ορίζοντα ΜΔΑ, είναι η πιθανότητα κάποιο φανταστικό γεγονός R να συμβεί στο τελευταίο βήμα μιας T-ορίζοντα ΜΔΑ, η οποία έχει το ίδιο δυναμικό μοντέλο με την άπειρου ορίζοντα ΜΔΑ.

Είδαμε πως σκοπός μας είναι να βρούμε τις βέλτιστες παραμέτρους, οι οποίες μεγιστοποιούν τη συνάρτηση ανταμοιβής:

$$J(\boldsymbol{\theta}) = \mathbb{E}\left\{\sum_{t=0}^{H} \gamma^t r_t; \boldsymbol{\theta}\right\}, \tag{2.5}$$

Διαφορετικά, αν $p_{\boldsymbol{\theta}}(\xi)$ δηλώνει την πιθανοφάνεια μιας πλήρους τροχιάς $\xi \in \Xi$, θα ισχύει:

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\xi}\{r(\xi)\} = \sum_{\Xi} p_{\boldsymbol{\theta}}(\xi) r(\xi) \tag{2.6}$$

Από τις παραπάνω εκφράσεις προκύπτει, πως αν ξ είναι μια τροχιά μήκους $|\xi| = T$ και $\mathcal{X}^T = \{\xi : |\xi| = T\}$ είναι ο χώρος όλων των μήκους T τροχιών, τότε η αναμενόμενη ανταμοιβή $J(\boldsymbol{\theta})$ του συστήματος είναι μια συνάρτηση πιθανοφάνειας μιας άπειρης μικτής κατανομής:

$$J(\boldsymbol{\theta}) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[r_t; \boldsymbol{\theta}] = \frac{1}{1-\gamma} \sum_{t=0}^{\infty} (1-\gamma)\gamma^t \int_{\xi \in \mathcal{X}^t} p(\xi|t, \boldsymbol{\theta}) p(R|\xi), \tag{2.7}$$

όπου $p(\xi|t, \boldsymbol{\theta})$ είναι η κατανομή όλων των t-μήκους τροχιών παραμετροποιημένη με παραμέτρους $\boldsymbol{\theta}$ και $p(R|\xi) = r_{|\xi|}$ είναι η πιθανότητα να συμβεί το R στο τελευταίο βήμα της τροχιάς ξ. Σύμφωνα με τη δυναμική της ΜΔΑ, η κατανομή όλων των μήκους T τροχιών, αν ακολουθούμε πολιτική $\pi_{\boldsymbol{\theta}}$ θα είναι:

$$p(\xi|T, \boldsymbol{\theta}) = p(x_{\xi 0}) \pi_{\boldsymbol{\theta}}(u_{\xi T}|x_{\xi T}) \prod_{t=0}^{T-1} p(x_{\xi(t+1)}|x_{\xi t}, u_{\xi t}) \pi_{\boldsymbol{\theta}}(x_{\xi t}|u_{\xi t}) \tag{2.8}$$

Αν το μοντέλο του συστήματος ήταν γνωστό τότε θα μπορούσαμε να μεγιστοποιήσουμε την αναμενόμενη ανταμοιβή $J(\boldsymbol{\theta})$ ως προς $\boldsymbol{\theta}$, χρησιμοποιώντας το μοντέλο (Toussaint and Storkey, 2006; Hoffman et al., 2008). Εδώ στηριζόμαστε στις πρόσφατες εργασίες (Vlassis and Toussaint, 2009; Vlassis et al., 2009), οι οποίες ορίζουν την ακόλουθη συνδυασμένη κατανομή πάνω στις τυχαίες μεταβλητές R, t, ξ και T:

$$p(R, t, \xi, T; \boldsymbol{\theta}) = \alpha(T) b(t) p(\xi|t; \boldsymbol{\theta}) p(R|\xi, T), \tag{2.9}$$

όπου η $p(\xi|t; \boldsymbol{\theta})$ δίνεται από την (2.8) και

$$\alpha(t) = (1-\delta)\delta^t, \tag{2.10}$$

$$b(t) = (1-\gamma/\delta)(\gamma/\delta)^t, \tag{2.11}$$

είναι γεωμετρικές κατανομές με $\gamma < \delta < 1$. Η μεταβλητή T μας δείχνει το μέγιστο μήκος τροχιάς, αφού μόνο τροχιές με μήκος $L \leq T$ είναι πιθανό να συμβούν.

Σύμφωνα με τους (Vlassis and Toussaint, 2009; Vlassis et al., 2009) η αναμενόμενη ανταμοιβή $J(\boldsymbol{\theta})$ είναι ανάλογη (proportional) της μικτής πιθανοφάνειας:

$$J(\boldsymbol{\theta}) \propto \sum_{T=0}^{\infty} \alpha(T) \sum_{t=0}^{\infty} b(t) \int_{\xi \in \mathcal{X}^t} p(\xi|t; \boldsymbol{\theta}) p(R|\xi, T), \tag{2.12}$$

με

$$p(R|\xi, T) = \begin{cases} r_{|\xi|}, & \text{αν } |\xi| \leq T \\ 0, & \text{αλλού} \end{cases} \tag{2.13}$$

Σκοπός μας είναι να μεγιστοποιήσουμε την αναμενόμενη ανταμοιβή $J(\boldsymbol{\theta})$ ως προς $\boldsymbol{\theta}$. Ισοδύναμα μπορούμε να μεγιστοποιήσουμε τη λογαριθμική πιθανοφάνεια (log likelihood) $L(\boldsymbol{\theta}) = \log p(R; \boldsymbol{\theta})$. Για να το πετύχουμε αυτό θα χρησιμοποιήσουμε τον αλγόριθμο ΕΜ, με τη βοήθεια του οποίου θα μεγιστοποιούμε επαναλαμβανόμενα μια συνάρτηση ενέργειας $F(\boldsymbol{\theta}, q)$, η οποία είναι κάτω όριο της

$L(\boldsymbol{\theta})$ (Neal and Hinton, 1998). Σε αντιστοιχία με την προηγούμενη ενότητα η $F(\boldsymbol{\theta}, q)$ είναι η $\mathcal{L}(q, \boldsymbol{\theta})$ και η $L(\boldsymbol{\theta})$ είναι η $\ln p(\boldsymbol{X}|\boldsymbol{\theta})$. Η F είναι μια συνάρτηση των παραμέτρων $\boldsymbol{\theta}$ και μιας τυχαίας κατανομής $q \equiv q(\xi, T, t)$ πάνω στις λανθάνουσες μεταβλητές $\xi \in \mathcal{X}$ και $t, T \in \mathbb{N}_0$, η οποία ορίζεται ως:

$$F(\boldsymbol{\theta}, q) = L(\boldsymbol{\theta}) - KL[q(\xi, T, t)||p(\xi, T, t|R; \boldsymbol{\theta}] = \tag{2.14}$$

$$= \mathbb{E}_{q(\xi,T,t)}[\log p(R, \xi, T, t; \boldsymbol{\theta}] + H(q) \tag{2.15}$$

Στην 2.14 υπολογίζουμε την Kullback - Leibler απόσταση ανάμεσα στην $q(\xi, T, t)$ και στην $p(\xi, T, t|R; \boldsymbol{\theta})$, η οποία είναι η εκ των υστέρων κατανομή Bayes στις t, ξ, T και στις παρατηρήσιμες μεταβλητές R και $\boldsymbol{\theta}$. Στην 2.15 ο πρώτος όρος είναι η αναμονή (expectation) της συνδυασμένης λογαριθμικής πιθανοφάνειας πάνω στην q και $H(q)$ είναι η εντροπία της q.

Όπως βλέπουμε από την 2.14, στο βήμα Ε του αλγορίθμου ΕΜ η βέλτιστη q^* πρέπει να ισούται με την εκ την υστέρων κατανομή Bayes με βάση τις παραμέτρους $\boldsymbol{\theta}_{old}$, το οποίο $\boldsymbol{\theta}_{old}$ έχει υπολογιστεί από το προηγούμενο βήμα Μ του ΕΜ. Έτσι θα ισχύει:

$$q^*(\xi, T, t) = p(\xi, T, t|R; \boldsymbol{\theta}_{old}) \tag{2.16}$$

$$\propto \alpha(T)b(t)p(\xi|t; \boldsymbol{\theta}_{old})p(R|\xi, T) \tag{2.17}$$

Σε αυτή την περίπτωση $F(\boldsymbol{\theta}, q^*) = L(\boldsymbol{\theta})$ και οι δύο συναρτήσεις εφάπτονται στο ύψος του $\boldsymbol{\theta}_{old}$.

Στο βήμα Μ του ΕΜ μεγιστοποιούμε την F ως προς $\boldsymbol{\theta}$ χρησιμοποιώντας την 2.15 και την βέλτιστη q^* από την 2.17 και έχουμε:

$$F(\boldsymbol{\theta}, q^*) = \mathbb{E}_{p(\xi,T,t|R;\boldsymbol{\theta}_{old})}[\log p(\xi|t; \boldsymbol{\theta}] \tag{2.18}$$

Από τη στιγμή που το μοντέλο της ΜΔΑ δεν είναι διαθέσιμο, δεν μπορούμε να υπολογίσουμε την παραπάνω ποσότητα αναλυτικά, οπότε πρέπει να την προσεγγίσουμε χρησιμοποιώντας δειγματικές τροχιές από την ΜΔΑ. Αυτή είναι και η γενική ιδέα του αλγορίθμου Monte Carlo Expectation Maximization (MCEM) (Tanner, 1990)· αντί να υπολογίσουμε την αναμονή (expectation) στο βήμα Ε αναλυτικά δειγματοληπτούμε από την q^*.

Για να πάρουμε ένα δείγμα από την $q^* \propto \alpha(T)b(t)p(\xi|t, \boldsymbol{\theta}_{old})p(R|\xi, T)$, αρχικά δειγματοληπτούμε ένα μέγιστο μήκος τροχιάς από την $\alpha(T)$, στη συνέχεια δειγματοληπτούμε μια τροχιά $\xi \sim p(\xi|T; \boldsymbol{\theta}_{old})$ μήκους T από την ΜΔΑ και στη συνέχεια χρησιμοποιούμε όλες τις t μήκους υποτροχιές της ξ, με $t = 0, 1, \ldots, T$.

Αν υποθέσουμε πως έχουμε m δειγματικές τροχιές $\xi_i, i = 1, 2, \ldots, m$ από την ΜΔΑ, τότε η εκτίμηση για την $F(\boldsymbol{\theta}, q^*)$ θα είναι:

$$\hat{F}(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{|\xi_i|+1}\sum_{t=0}^{|\xi_i|} w_{it}\log p(\xi_i|t; \boldsymbol{\theta}), \tag{2.19}$$

όπου $p(\xi_i|t; \boldsymbol{\theta})$ είναι η πιθανοφάνεια του μήκους t αρχικού τμήματος της ξ_i και $w_{it} = b(t)r_{it}$, με r_{it} να είναι η ανταμοιβή που έλαβε το σύστημα την χρονική στιγμή t της τροχιάς ξ_i.

Όταν η $\log(\xi|t;\boldsymbol{\theta})$ είναι τετραγωνική (quadratic) συνάρτηση ως προς τις παραμέτρους $\boldsymbol{\theta}$ (όπως για παράδειγμα συμβαίνει αν ο ελεγκτής είναι γραμμικός - Gauss), τότε η μεγιστοποίησης της 2.19 μπορεί να γίνει αναλυτικά.

Χρησιμοποιούμε τον ίδιο ελεγκτή με τους (Kober and Peters, 2008; Ruckstie et al., 2008):

$$u_t = (\boldsymbol{\theta} + \boldsymbol{\varepsilon}_t)\boldsymbol{\phi}(\boldsymbol{x}_t), \tag{2.20}$$

όπου $\boldsymbol{\phi} : \mathbb{R}^n \mapsto \mathbb{R}^d$ είναι σταθερές συναρτήσεις βάσης και ε_t είναι λευκός Gaussian θόρυβος $\varepsilon_t \sim \mathcal{N}(\varepsilon_t; 0, \sigma^2 I_d)$, ο οποίος χρησιμοποιείται για την εξερεύνηση του χώρου των παραμέτρων. Το σ μπορεί να είναι σταθερό, να μεταβάλλεται σε κάθε επανάληψη του αλγορίθμου ή να είναι κι αυτό μια από τις παραμέτρους προς βελτιστοποίηση.

Για μια μόνο τροχιά ξ_i η 2.19 γίνεται:

$$\hat{F}(\boldsymbol{\theta}) = \frac{1}{\xi_i + 1} \sum_{\tau=0}^{|\xi_i|} b(\tau) r_{i\tau} \sum_{t=0}^{\tau} \log \pi_{\boldsymbol{\theta}}(u_{it}|x_{it}) = \frac{1}{\xi_i + 1} \sum_{t=0}^{|\xi_i|} \left[\sum_{\tau=t}^{|\xi_i|} b(\tau) r_{i\tau} \right] \log \pi_{\boldsymbol{\theta}}(u_{it}|x_{it}) = \tag{2.21}$$

$$= \frac{1}{\xi_i + 1} \sum_{t=0}^{|\xi_i|} Q_{it} \log \pi_{\boldsymbol{\theta}}(u_{it}|x_{it}), \tag{2.22}$$

όπου

$$Q_{it} = \sum_{\tau=t}^{|\xi_i|} b(\tau) r_{i\tau} \tag{2.23}$$

Παραγωγίζοντας την 2.22 ως προς $\boldsymbol{\theta}$ έχουμε τελικά:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \frac{\sum_{i=1}^{m} \frac{1}{|\xi_i|+1} \sum_{t=0}^{|\xi_i|} Q_{it} \boldsymbol{\varepsilon}_{it}}{\sum_{i=1}^{m} \frac{1}{|\xi_i|+1} \sum_{t=0}^{|\xi_i|} Q_{it}} \tag{2.24}$$

Συνοπτικά ο αλγόριθμος MCEM φαίνεται στον πίνακα 2.1.

2.3 Ο αλγόριθμος Policy learning by Weighting Exploration with the Returns (PoWER)

Αν στον αλγόριθμο MCEM θέσουμε $\alpha(T) = 1$ για $T = H$ και 0 για οποιοδήποτε άλλο μήκος τροχιάς και επιπλέον θέσουμε $b(t) = 1/(1 + H)$ για $t = 1, 2, \ldots, H$, τότε η εξίσωση 2.24 παραμένει αμετάβλητη, με τη μόνη απλοποίηση ότι θα έχουμε το ίδιο μήκος H για όλα τα επεισόδια και ο όρος $b(t)$ θα είναι μια σταθερά, οπότε $Q_{it} = \sum_{t=\tau}^{H} r_{it}$. Ο νέος αλγόριθμος που προκύπτει είναι ο PoWER των (Kober and Peters, 2008).

Ο αλγόριθμος PoWER είναι σχεδιασμένος για προβλήματα πεπερασμένου ορίζοντα, ενώ ο αλγόριθμος MCEM είναι δομημένος έτσι ώστε να ανταποκρίνεται σε προβλήματα άπειρου ορίζοντα στα

Πίνακας 2.1: Μάθηση πολιτικής με τον αλγόριθμο MCEM

Είσοδος: αρχικές παράμετροι πολιτικής $\boldsymbol{\theta}$

Επανάλαβε
  Επανάλαβε για $i = 1 : m$
    Επίλεξε ένα τυχαίο μήκος τροχιάς T_i από κατάλληλη γεωμετρική κατανομή
    Συνέλεξε δειγματική τροχιά με $\boldsymbol{u}_t = (\boldsymbol{\theta} + \boldsymbol{\varepsilon}_t)^T \boldsymbol{\phi}(\boldsymbol{x}_t)$, όπου $[\boldsymbol{\varepsilon}_t]_{ij} \sim \mathcal{N}(0, \sigma_{ij}^2)$
    και αποθήκευσε όλα τα $(t, \boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{x}_{t+1}, \boldsymbol{\varepsilon}_t, r_{t+1})$ για $t = 1, 2, \ldots, T_i$
  Έως $i = m$
  Υπολόγισε τα $Q_{it} = \sum_{t=\tau}^{T_i} b(t) r_{it}$
  Βρες νέες παραμέτρους $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + (\sum_{i=1}^{m} \frac{1}{|\xi_i|+1} \sum_{t=0}^{|\xi_i|} Q_{it} \boldsymbol{\varepsilon}_{it}) / (\sum_{i=1}^{m} \frac{1}{|\xi_i|+1} \sum_{t=0}^{|\xi_i|} Q_{it})$
Έως $\boldsymbol{\theta}_{k+1} \simeq \boldsymbol{\theta}_k$

Έξοδος: θ_k

οποία η ανταμοιβή του συστήματος μειώνεται με το χρόνο. Αυτό δεν σημαίνει πως ο PoWER δεν μπορεί να επιλύσει προβλήματα άπειρου ορίζοντα, απλά απαιτεί την εύρεση κατάλληλου μήκους τροχιάς H. Το πρόβλημα είναι πως δεν υπάρχει κάποιος κανόνας για το βέλτιστο μήκος H μπορεί να διαφέρει αρκετά από πρόβλημα σε πρόβλημα, αλλά και για διαφορετικές αρχικές τιμές του ίδιου προβλήματος.

Η δεύτερη διαφορά ανάμεσα στον MCEM και στον PoWER είναι οι ποσότητες $b(t)$ οι οποίες μειώνονται με το χρόνο, δίνοντας μεγαλύτερο βάρος στις ανταμοιβές που λαμβάνονται στα πρώτα βήματα της τροχιάς. Το γεγονός αυτό μπορεί να βοηθήσει σημαντικά στη σύγκλιση του MCEM όταν η δυναμική της ΜΔΑ έχει αρκετό θόρυβο, καθώς στην περίπτωση αυτή οι ανταμοιβές που λαμβάνει το σύστημα σε βάθος χρόνου έχουν εκφυλιστεί από το θόρυβο του περιβάλλοντος.

Συνοπτικά ο αλγόριθμος PoWER φαίνεται στον πίνακα 2.2.

Πίνακας 2.2: Μάθηση πολιτικής με τον αλγόριθμο PoWER

Είσοδος: αρχικές παράμετροι πολιτικής $\boldsymbol{\theta}$

Επανάλαβε
  Συνέλεξε m δειγματικές τροχιές με $\boldsymbol{u}_t = (\boldsymbol{\theta} + \boldsymbol{\varepsilon}_t)^T \boldsymbol{\phi}(\boldsymbol{x}_t)$, όπου $[\boldsymbol{\varepsilon}_t]_{ij} \sim \mathcal{N}(0, \sigma_{ij}^2)$
  και αποθήκευσε όλα τα $(t, \boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{x}_{t+1}, \boldsymbol{\varepsilon}_t, r_{t+1})$ για $t = 1, 2, \ldots, H$
  Υπολόγισε τα $\hat{Q}_{it}^{\pi}(\boldsymbol{x}, \boldsymbol{u}, \tau) = \sum_{t=\tau}^{T_i} r_{it}$
  Βρες νέες παραμέτρους $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + (\sum_{i=1}^{m} \sum_{t=0}^{H} Q_{it}(\boldsymbol{x}, \boldsymbol{u}, t) \boldsymbol{\varepsilon}_{it}) / (\sum_{i=1}^{m} \sum_{t=0}^{H} Q_{it}(\boldsymbol{x}, \boldsymbol{u}, t))$
Έως $\boldsymbol{\theta}_{k+1} \simeq \boldsymbol{\theta}_k$

Έξοδος: θ_k

2.4 Μέθοδοι Επιτάχυνσης των Αλγορίθμων Πιθανοτικού Συμπερασμού

Μετά από μια σειρά πειραμάτων που πραγματοποιήθηκαν χρησιμοποιώντας τους αλγορίθμους του κεφαλαίου αυτού, διαπιστώθηκε η σχετικά αργή τους σύγκλιση. Για το λόγο αυτό διερευνήθηκαν διάφορες μέθοδοι για την επιτάχυνσή τους.

2.4.1 Χρήση των καλύτερων δειγματικών τροχιών

Η μέθοδος αυτή χρησιμοποιείται στους Cross Entropy (Rubinstein and Kroese, 2004; Mannor et al., 2003; Chang et al., 2007) αλγορίθμους. Σε αυτού του είδους τους αλγορίθμους λαμβάνονται m δειγματικές τροχιές από το σύστημα με πολιτική $u_t = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{x}_t) + \varepsilon_t$, με $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, καθεμιά από τις οποίες συλλέγει συνολική ανταμοιβή $R_i, i = 1, 2, \ldots, m$. Στη συνέχεια οι τροχιές ταξινομούνται με βάση τις ανταμοιβές τους από την καλύτερη προς τη χειρότερη και για τον υπολογισμό των νέων παραμέτρων $\boldsymbol{\theta}_{k+1}$ χρησιμοποιείται ένα ποσοστό p από τις τροχιές με τη μεγαλύτερη συνολική ανταμοιβή. Με τον τρόπο αυτό δεν συνυπολογίζονται στην εύρεση των νέων παραμέτρων ελεγκτές που προέκυψαν από εξερεύνηση του χώρου καταστάσεων μακριά από το βέλτιστο, ακόμα και αν η συνεισφορά τους στις νέες παραμέτρους θα ήταν μικρή (λόγω της μικρής ανταμοιβής που θα είχαν συλλέξει). Επιπλέον, δεν συνυπολογίζονται και τροχιές, οι οποίες δεν συνέλεξαν μεγάλη ανταμοιβή λόγω της στοχαστικότητας του συστήματος.

Στη δική μας περίπτωση, για τους αλγορίθμους MCEM και PoWER χρησιμοποιούμε ακριβώς την ίδια διαδικασία με τους Cross Entropy αλγορίθμους, αλλά ακολουθώντας την πολιτική της σχέσης 2.20 για τη συλλογή τροχιών του συστήματος. Ο αλγόριθμος PoWER που προκύπτει φαίνεται στον πίνακα 2.3 και ο αντίστοιχος MCEM στον πίνακα 2.4.

Πίνακας 2.3: Μάθηση πολιτικής με τον αλγόριθμο PoWER με χρήση των καλύτερων δειγματικών τροχιών

Είσοδος: αρχικές παράμετροι πολιτικής $\boldsymbol{\theta}_0$

Επανάλαβε

Συνέλεξε m δειγματικές τροχιές με $\boldsymbol{u}_t = (\boldsymbol{\theta} + \varepsilon_t)^T \boldsymbol{\phi}(\boldsymbol{x}_t)$, όπου $[\varepsilon_t]_{ij} \sim \mathcal{N}(0, \sigma_{ij}^2)$ και αποθήκευσε όλα τα $(t, \boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{x}_{t+1}, \varepsilon_t, r_{t+1})$ για $t = 1, 2, \ldots, H$

Υπολόγισε όλα τα $R_i = \sum_{t=0}^{H} r_{it}, i = 1, 2, \ldots, m$
Ταξινόμησε τις τροχιές με βάση τη συνολική ανταμοιβή που συνέλεξαν
Υπολόγισε τα $\hat{Q}_{it}^{\pi}(\boldsymbol{x}, \boldsymbol{u}, \tau) = \sum_{t=\tau}^{T_i} r_{it}$ *για τις* $p\%$ *καλύτερες τροχιές*

Βρες νέες παραμέτρους $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + (\sum_{i=1}^{pm} \sum_{t=0}^{H} Q_{it}(\boldsymbol{x}, \boldsymbol{u}, t) \varepsilon_{it}) / (\sum_{i=1}^{pm} \sum_{t=0}^{H} Q_{it}(\boldsymbol{x}, \boldsymbol{u}, t))$
Έως $\boldsymbol{\theta}_{k+1} \simeq \boldsymbol{\theta}_k$

Έξοδος: $\boldsymbol{\theta}_k$

Πίνακας 2.4: Μάθηση πολιτικής με τον αλγόριθμο MCEM με χρήση των καλύτερων δειγματικών τροχιών

Είσοδος: αρχικές παράμετροι πολιτικής $\boldsymbol{\theta}_0$

Επανάλαβε
  Επανάλαβε για $i = 1 : m$
    Επίλεξε ένα τυχαίο μήκος τροχιάς T_i από κατάλληλη γεωμετρική κατανομή
    Συνέλεξε δειγματική τροχιά με $\boldsymbol{u}_t = (\boldsymbol{\theta} + \boldsymbol{\varepsilon}_t)^T \boldsymbol{\phi}(\boldsymbol{x}_t)$, όπου $[\boldsymbol{\varepsilon}_t]_{ij} \sim \mathcal{N}(0, \sigma_{ij}^2)$
    και αποθήκευσε όλα τα $(t, \boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{x}_{t+1}, \boldsymbol{\varepsilon}_t, r_{t+1})$ για $t = 1, 2, \ldots, T_m$
  Έως $i = m$

  Υπολόγισε όλα τα $R_i = \left(\sum_{t=0}^{T_i} r_{it}\right) / T_i, i = 1, 2, \ldots, m$
  Ταξινόμησε τις τροχιές με βάση τη συνολική ανταμοιβή που συνέλεξαν
  Υπολόγισε τα $Q_{it} = \sum_{t=\tau}^{T_i} b(t) r_{it}$ *για τις* $p\%$ *καλύτερες τροχιές*

  Βρες νέες παραμέτρους $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + (\sum_{i=1}^{pm} \frac{1}{|\xi_i|+1} \sum_{t=0}^{|\xi_i|} Q_{it} \boldsymbol{\varepsilon}_{it}) / (\sum_{i=1}^{pm} \frac{1}{|\xi_i|+1} \sum_{t=0}^{|\xi_i|} Q_{it})$
Έως $\boldsymbol{\theta}_{k+1} \simeq \boldsymbol{\theta}_k$

Έξοδος: $\boldsymbol{\theta}_k$

2.4.2 Αναζήτηση Επί Γραμμής (Line Search)

Οι αλγόριθμοι MCEM και PoWER σε κάθε επανάληψή τους υπολογίζουν ένα νέο σετ παραμέτρων $\boldsymbol{\theta}_{k+1}$. Αντίθετα, όπως είδαμε, οι αλγόριθμοι πολιτικής κλίσης υπολογίζουν μια νέα κατεύθυνση του διανύσματος κλίσης $\boldsymbol{g}$ και οι νέες παράμετροι προκύπτουν από τη σχέση $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha_k \boldsymbol{g}$.

Οι αλγόριθμοι πολιτικής κλίσης μπορούν να δώσουν καλύτερα αποτελέσματα αν συνδυαστούν με τεχνικές αναζήτησης επί γραμμής (κυρίως αν το σύστημα δεν είναι στοχαστικό) (Bertsekas et al., 1995; Bazaraa et al., 2006). Αφού έχει υπολογιστεί το διάνυσμα κλίσης μπορούμε να αναζητήσουμε πάνω στην κατεύθυνση του μια καλύτερη λύση, παράγοντας συνεχώς καινούριες παραμέτρους με τη σχέση:

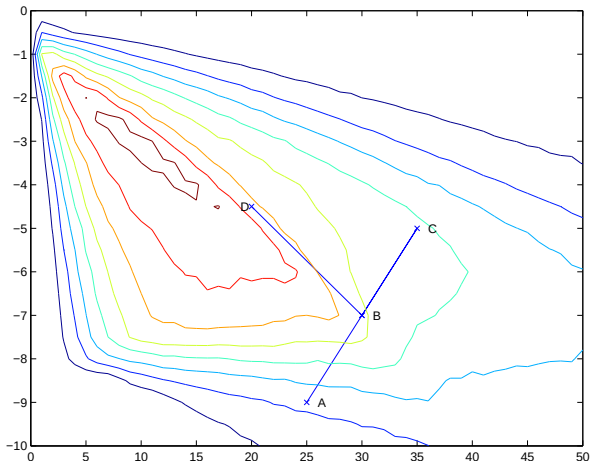
$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \mu \alpha_k \boldsymbol{g}, \tag{2.25}$$

όπου μ μια σταθερά. Η διαδικασία σταματά όταν κάποια λύση είναι χειρότερη από την προηγούμενη και το διάνυσμα των παραμέτρων θα είναι η καλύτερη λύση που βρήκαμε σε όλη τη διαδικασία, όπως φαίνεται και στην εικόνα 2.3.

Με βάση τα παραπάνω, μπορούμε να θεωρήσουμε πως οι MCEM και PoWER δεν υπολογίζουν απ' ευθείας ένα νέο σετ παραμέτρων, αλλά υπολογίζουν την ποσότητα που θα χρησιμοποιηθεί στην αναζήτηση επί γραμμής, δηλαδή:

$$\mu \alpha_k \boldsymbol{g} = \boldsymbol{\theta}_{k+1}^{MCEM} - \boldsymbol{\theta}_k^{MCEM} \tag{2.26}$$

Σε αυτήν την περίπτωση μένει να οριστεί πώς θα αποφασίζεται αν μια λύση που προέκυψε από την αναζήτηση επί γραμμής είναι καλύτερη ή χειρότερη από την προηγούμενη. Ο πιο εύκολος



Σχήμα 2.3: Μέθοδος αναζήτησης επί γραμμής: το A είναι η αρχική λύση, το B είναι μια η λύση που προέκυψε από την αναζήτηση επί γραμμής, το C είναι μια λύση χειρότερη από το B και το D είναι η λύση που προέκυψε από την νέα εκτέλεση του αλγορίθμου με αρχική τιμή το B

τρόπος, ο οποίος αποδεικνύεται αποδοτικός στην πράξη είναι να εκτελέσουμε το σύστημα για έναν συγκεκριμένο αριθμό τροχιών n με $n \ll m$ σταθερού μήκους H, όπου m (πίνακας 2.2) ο αριθμός των τροχιών που εκτελούνται σε κάθε επανάληψη του αλγορίθμου και να θεωρήσουμε σαν αναμενόμενη ανταμοιβή του κάθε σετ παραμέτρων το μέσο όρο των ανταμοιβών που συνέλεξε στις n τροχιές το κάθε σετ.

Τελικά στους αλγορίθμους MCEM και PoWER θα προστεθεί η μέθοδος που φαίνεται στον πίνακα 2.5.

Πίνακας 2.5: Μέθοδος Αναζήτησης επί Γραμμής για τους MCEM και PoWER

Είσοδος: αρχικές παράμετροι πολιτικής $\boldsymbol{\theta}_k$, βήμα $\boldsymbol{D}$ //$\boldsymbol{D} = \boldsymbol{\theta}_k - \boldsymbol{\theta}_{k-1}$

Επανέλαβε
- $c = 1$
- Συνέλεξε $n \ll m$ δειγματικές τροχιές με $\boldsymbol{u}_t = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{x}_t)$
- Υπολόγισε το $R(c) = \left(\sum_{i=1}^{n} \sum_{t=0}^{H} r_{it} \right) / n$
- $c = c + 1$
- Αν $c > 1$
  - θέσε $\boldsymbol{\theta}_k = \boldsymbol{\theta}_k + \boldsymbol{D}$
- Τέλος Αν

Έως $R(c-1) > R(c-2)$

Έξοδος: $\boldsymbol{\theta}_k - 2 * \boldsymbol{D}$

Κεφάλαιο 3

Πειράματα

3.1 Διδιάστατο Συνθετικό Πρόβλημα

Το πρώτο πρόβλημα είναι μια ΜΔΑ δύο διαστάσεων με λίγο θόρυβο στο δυναμικό μοντέλο, παρόμοιο με αυτό που χρησιμοποιούν οι (Toussaint and Storkey, 2006). Το συγκεκριμένο πρόβλημα έχει μόνο ένα μέγιστο, οπότε αποτελεί ιδανικό benchmark για την ταχύτητα και την ποιότητα σύγκλισης διαφορετικών αλγορίθμων. Ο χώρος των καταστάσεων είναι $\boldsymbol{x} = [x_1, x_2]$, όπου x_1 είναι η θέση του ρομπότ και x_2 είναι η ταχύτητα του ρομπότ, η ενέργεια ελέγχου u είναι η επιτάχυνση, και το δυναμικό μοντέλο της ΜΔΑ είναι:

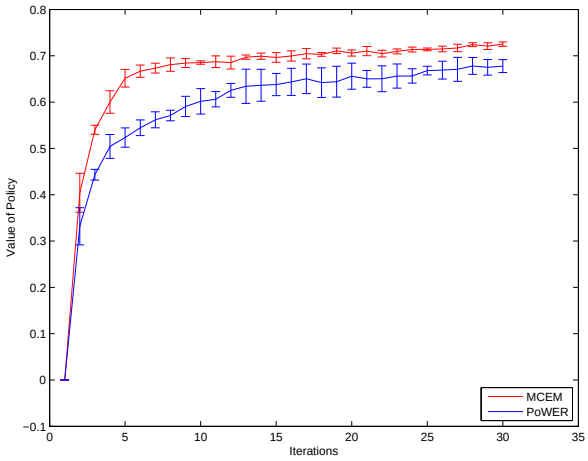
$$x_2(t+1) = x_2(t) - u(t) + \kappa, \tag{3.1}$$

$$x_1(t+1) = x_1(t) - 0.1x_2(t+1) + \kappa, \tag{3.2}$$

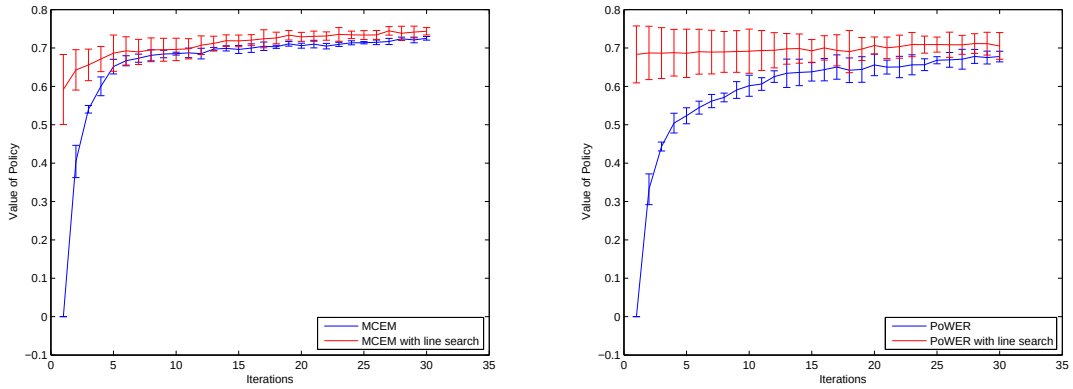
όπου κ είναι λευκός Gaussian θόρυβος με μέση τιμή μηδέν και $\sigma_\kappa = 0.003$. Το ρομπότ εκκινεί από τη θέση $x_1 = 1$ (συν κάποιο Gaussian θόρυβο), έχει αρχική ταχύτητα $x_2 = 0$ και πρέπει να φτάσει στην κατάσταση $[0, 0]$. Η ανταμοιβή εδώ είναι $r(t) = \exp(-0.5||\boldsymbol{x}(t)||_2/(0.03)^2)$. Για να περιορίσουμε την τιμή της επιτάχυνσης σε κάποιο εύρος (διαφορετικά η λύση είναι απλή) παραμετροποιούμε την πολιτική ελέγχου με μια σιγμοειδή $u_t = 1/[1 + \exp(-(\boldsymbol{\theta} + \boldsymbol{\varepsilon}_t)^T \boldsymbol{x})] - 0.5$. Στην εικόνα 3.1 φαίνονται οι καμπύλες μάθησης των MCEM και PoWER. Στον MCEM χρησιμοποιήσαμε $\gamma = 0.95$, $\delta = 0.99$ και μέγιστο μήκος τροχιάς $T = 100$ βήματα, ενώ στον PoWER χρησιμοποιήσαμε σταθερό μήκος τροχιάς $H = 100$ βήματα. Σε κάθε επανάληψη του αλγορίθμου συλλέγουμε 50 τροχιές.

Στη συνέχεια δοκιμάσαμε τους αλγορίθμους με τις ίδιες ρυθμίσεις με πριν, αλλά με την προσθήκη αναζήτησης επί γραμμής. Για την εκτίμηση της αξίας κάθε πολιτικής στην αναζήτηση επί γραμμής συλλέγαμε τα δεδομένα δέκα τροχιών στο σύστημα. Στην εικόνα 3.2 παρουσιάζεται η σύγκριση των δύο αλγορίθμων με και χωρίς τη χρήση αναζήτησης επί γραμμής. Παρατηρούμε πως η μέθοδος βελτιώνει θεαματικά τον PoWER, ο οποίος έχει πλέον συγκρίσιμα αποτελέσματα με τον MCEM.

Τέλος, διατηρώντας τις ίδιες ρυθμίσεις, προσθέσαμε στους αλγορίθμους τη μέθοδο χρήσης των καλύτερων τροχιών, διατηρώντας μόνο τις 2 καλύτερες τροχιές σε κάθε επανάληψη των αλγορίθμων. Τα αποτελέσματα και για τους δύο αλγορίθμους φαίνονται στην εικόνα 3.3, όπου παρατηρούμε σημαντική βελτίωση στην ταχύτητα σύγκλισης των αλγορίθμων.



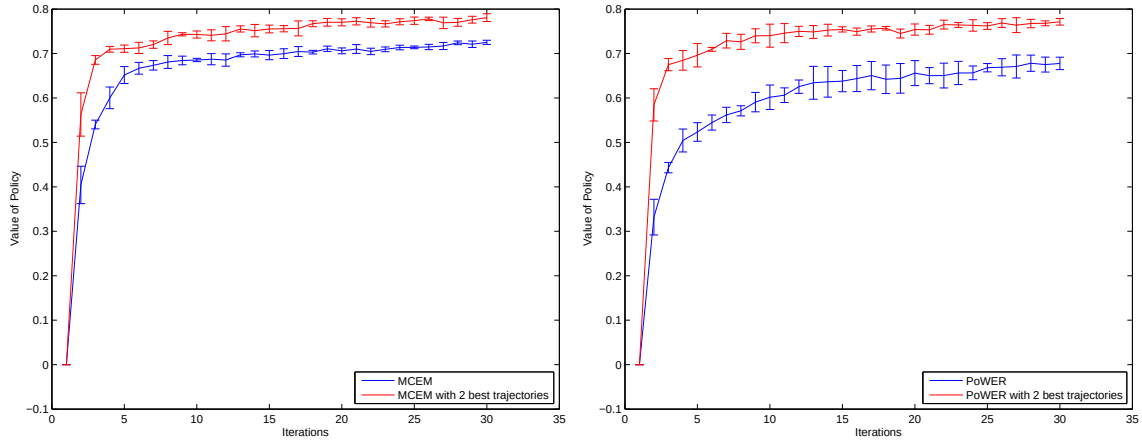
Σχήμα 3.1: Καμπύλες μάθησης των αλγορίθμων MCEM και PoWER στο διδιάστατο συνθετικό πρόβλημα με προσθήκη μικρής ποσότητας θορύβου



Σχήμα 3.2: Αριστερά: ο αλγόριθμος MCEM με και χωρίς αναζήτηση επί γραμμής. Δεξιά: ο αλγόριθμος PoWER με και χωρίς αναζήτηση επί γραμμής

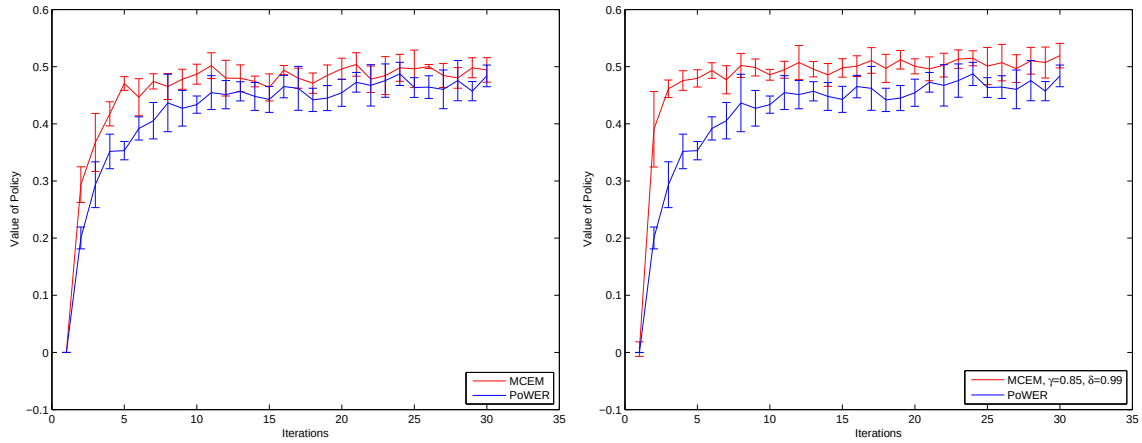
Στη συνέχεια δοκιμάσαμε τους ίδιους αλγορίθμους σε ένα πρόβλημα με περισσότερο θόρυβο ($\sigma_\kappa = 0.01$) και πραγματοποιήσαμε την ίδια ακριβώς σειρά πειραμάτων με προηγουμένως. Έτσι, οι καμπύλες μάθησης των δύο αλγορίθμων φαίνονται στην εικόνα 3.4(a). Παρατηρούμε πως οι δύο αλγόριθμοι αρχίζουν να εμφανίζουν παρόμοια αποτελέσματα, λόγω της παρουσίας θορύβου. Ο PoWER δεν μπορεί να αντιμετωπίσει το φαινόμενο του θορύβου, ενώ ο MCEM έχει τη δυνατότητα να δώσει μεγαλύτερο βάρος στις ανταμοιβές που λαμβάνει στα πρώτα βήματα του επεισοδίου, αντισταθμίζοντας έτσι την επίδραση του υψηλού θορύβου. Αυτό ακριβώς φαίνεται στην εικόνα 3.4(b), όπου ο MCEM έχει παραμέτρους $\gamma = 0.85$ και $\delta = 0.99$.

Οι καμπύλες μάθησης των αλγορίθμων με αναζήτηση επί γραμμής φαίνονται στην εικόνα 3.5 και οι καμπύλες μάθησης των αλγορίθμων με χρήση των 2 καλύτερων τροχιών στην εικόνα 3.6. Παρατηρούμε μεγάλη βελτίωση στους αλγορίθμους με χρήση αναζήτησης επί γραμμής και σημαντικά



Σχήμα 3.3: Αριστερά: ο αλγόριθμος MCEM με και χωρίς χρήση της μεθόδου καλύτερων τροχιών. Δεξιά: ο αλγόριθμος PoWER με και χωρίς χρήση της μεθόδου καλύτερων τροχιών

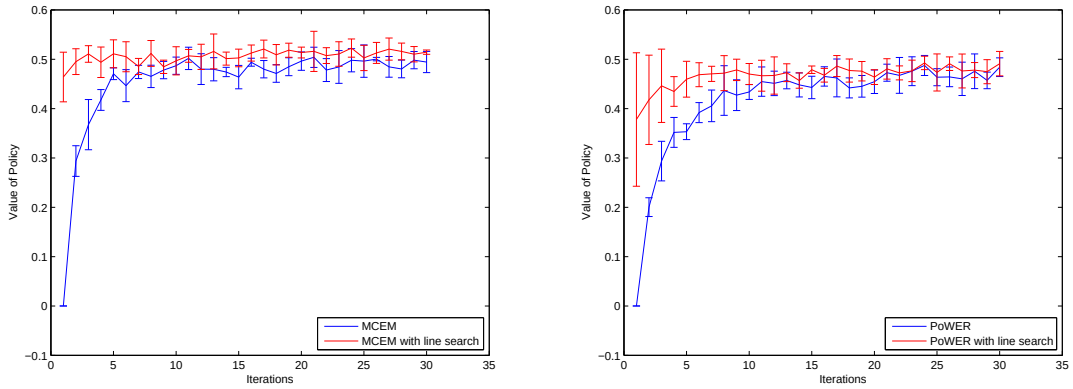
μικρότερη στους αλγορίθμους που κάνουν χρήση των καλύτερων τροχιών.



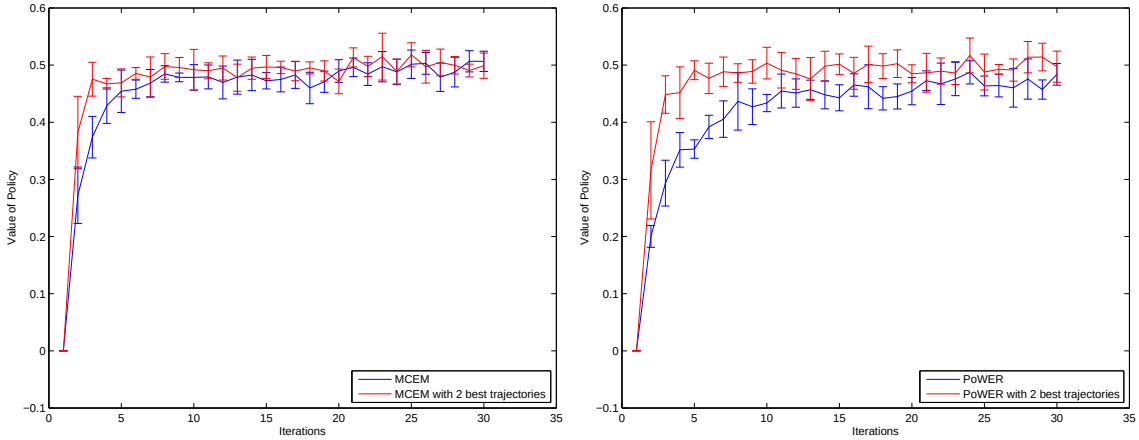
Σχήμα 3.4: Αριστερά: Καμπύλες μάθησης των αλγορίθμων MCEM και PoWER στο διδιάστατο συνθετικό πρόβλημα με προσθήκη σημαντικής ποσότητας θορύβου. Δεξιά: Οι καμπύλες μάθησης όταν ο MCEM έχει παραμέτρους $\gamma = 0.85$ και $\delta = 0.99$

Τέλος, πραγματοποιήσαμε τις ίδιες δοκιμές και για αρκετά αυξημένα επίπεδα θορύβου $\sigma_\kappa = 0.03$. Τα αποτελέσματα φαίνονται στις εικόνες 3.7(a) για MCEM με $\gamma = 0.95$ και $\delta = 0.99$ και 3.7(b) για $\gamma = 0.85$ και $\delta = 0.99$.

Οι καμπύλες μάθησης των αλγορίθμων με αναζήτηση επί γραμμής φαίνονται στην εικόνα 3.8. Βλέπουμε πως η μεγάλη ποσότητα θορύβου έχει επηρεάσει την ταχύτητα σύγκλισης των αλγορίθμων με αναζήτηση επί γραμμής, αφού είναι δύσκολο να γίνει ασφαλής εκτίμηση της ποιότητας της κάθε λύσης. Παρόμοια αποτελέσματα είχαμε και με χρήση των καλύτερων τροχιών στους δύο αλγορίθμους και γι' αυτό δεν παρατίθενται.

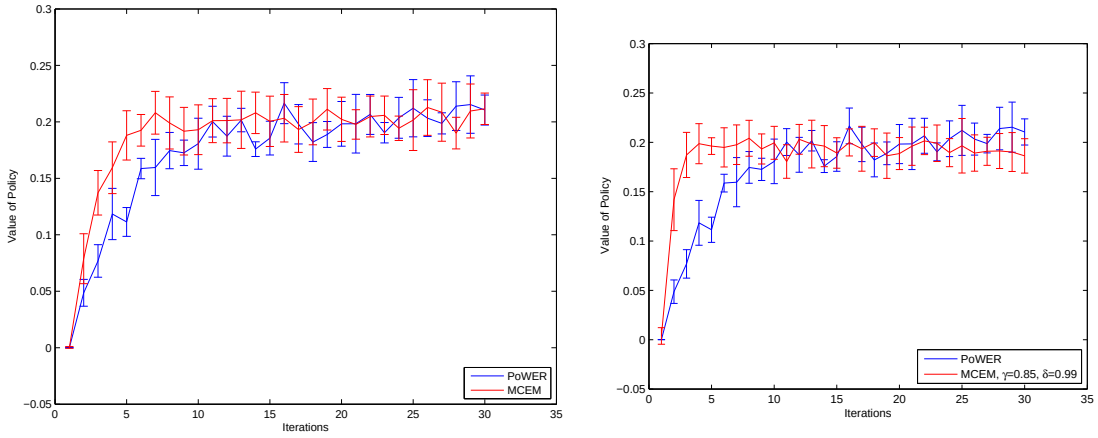


Σχήμα 3.5: Αριστερά: ο αλγόριθμος MCEM με και χωρίς αναζήτηση επί γραμμής. Δεξιά: ο αλγόριθμος PoWER με και χωρίς αναζήτηση επί γραμμής

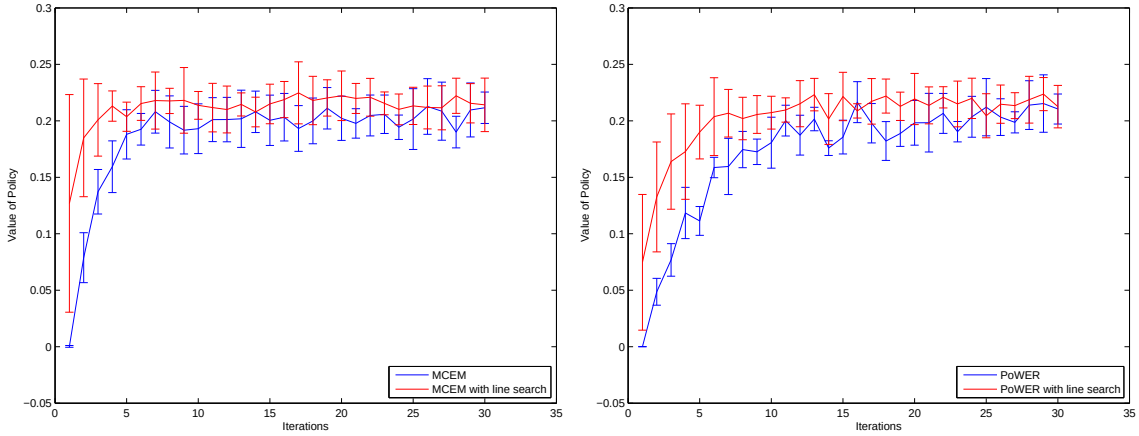


Σχήμα 3.6: Αριστερά: ο αλγόριθμος MCEM με και χωρίς χρήση της μεθόδου καλύτερων τροχιών για σημαντική προσθήκη θορύβου. Δεξιά: ο αλγόριθμος PoWER με και χωρίς χρήση της μεθόδου καλύτερων τροχιών

Επειδή σε συνθήκες υψηλού θορύβου είναι πολύ δύσκολο να εκτιμήσουμε με ακρίβεια την ποιότητα της λύσης που παράγει ο κάθε αλγόριθμος, με αποτέλεσμα η αναμενόμενη ανταμοιβή να φαίνεται η ίδια και με τη χρήση όλων των τροχιών και με τη χρήση των καλύτερων μόνο (π.χ. εικόνα 3.6), τροποποιήσαμε λίγο το διδιάστατο πρόβλημα δυσκολεύοντάς το, ώστε να μελετήσουμε τη συμπεριφορά των αλγορίθμων, ως προς την εύρεση βέλτιστης λύσης με τη χρήση των καλύτερων τροχιών. Έτσι, διατηρούμε ίδια τη δυναμική του προβλήματος, αλλά μεταβάλλουμε το θόρυβο και τη συνάρτηση ανταμοιβής ως εξής:



Σχήμα 3.7: Αριστερά: Καμπύλες μάθησης των αλγορίθμων MCEM και PoWER στο διδιάστατο συνθετικό πρόβλημα με προσθήκη μεγάλης ποσότητας θορύβου. Δεξιά: Οι καμπύλες μάθησης όταν ο MCEM έχει παραμέτρους $\gamma = 0.85$ και $\delta = 0.99$



Σχήμα 3.8: Αριστερά: ο αλγόριθμος MCEM με και χωρίς αναζήτηση επί γραμμής. Δεξιά: ο αλγόριθμος PoWER με και χωρίς αναζήτηση επί γραμμής

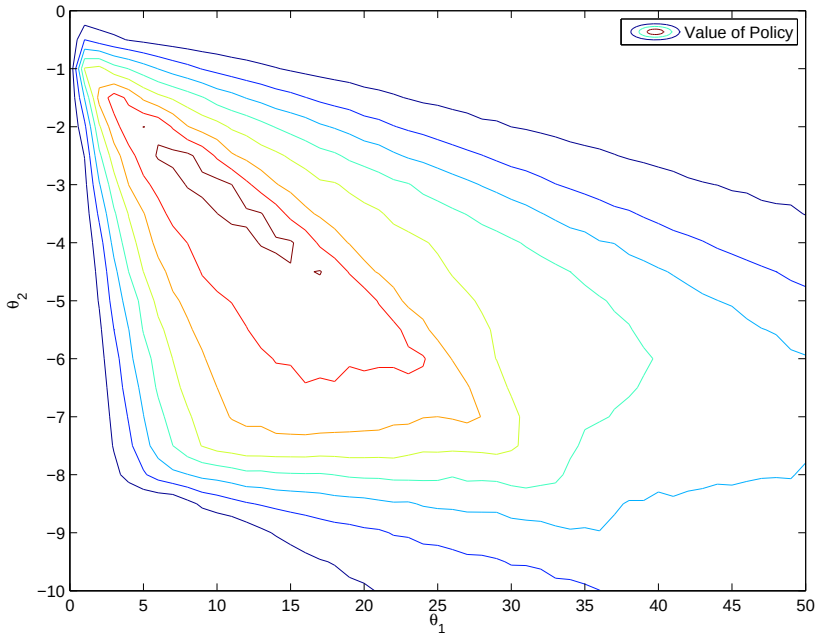
$$\sigma_\kappa = 0.02, \tag{3.3}$$

$$r(t) = \begin{cases} 1, & \text{αν } ||\boldsymbol{x}||_2 < 0.1 \\ 0, & \text{αλλού} \end{cases} \tag{3.4}$$

Τώρα ο ελεγκτής λαμβάνει ανταμοιβή μόνο όταν το ρομπότ εισέλθει σε μια συγκεκριμένη περιοχή του χώρου καταστάσεων.

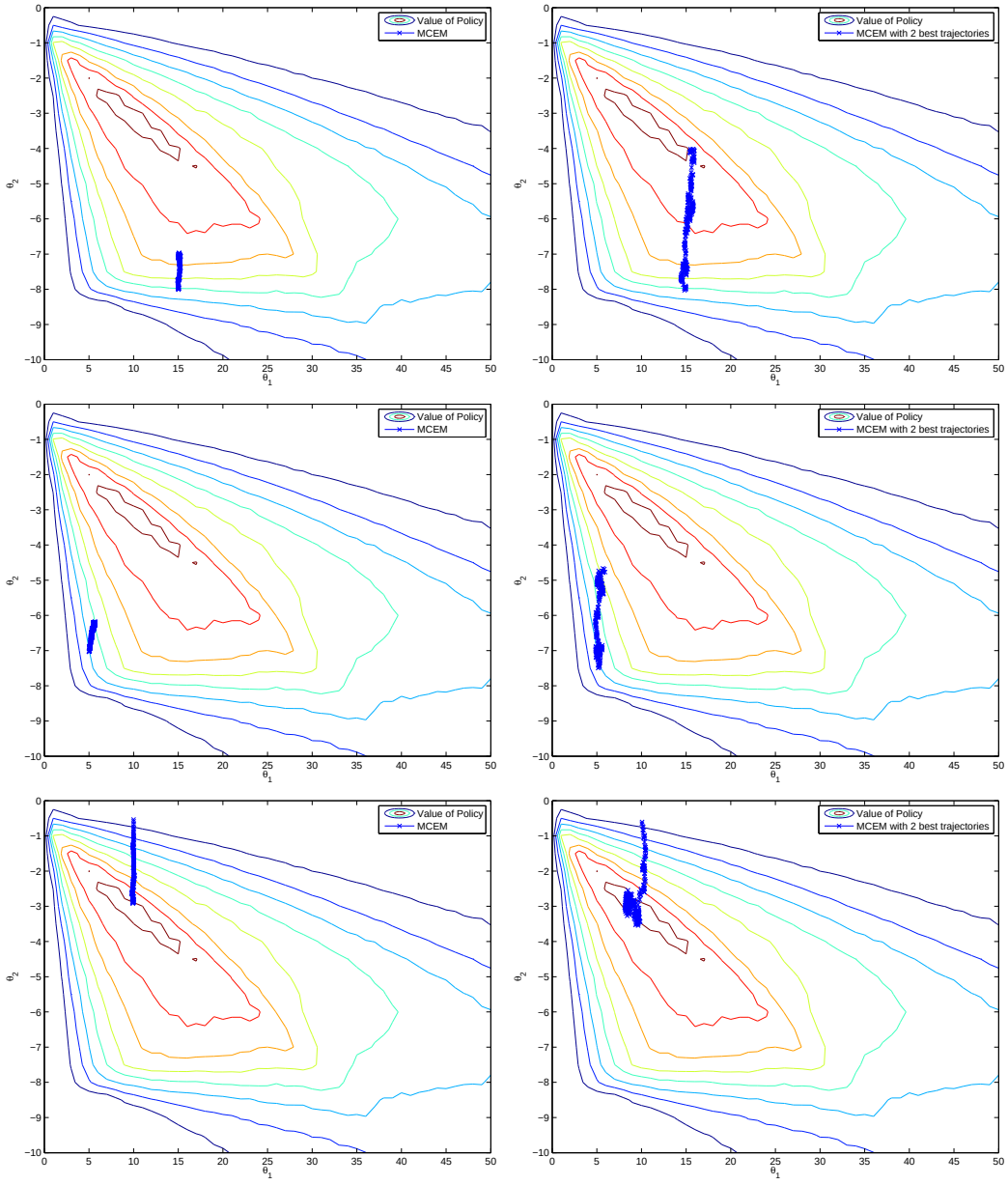
Θα μελετήσουμε τη συμπεριφορά μόνο του MCEM (αφού ανάλογα είναι τα αποτελέσματα και για τον PoWER), οπότε θεωρούμε ότι οι μελλοντικές ανταμοιβές που συλλέγονται από το σύστημα

θα υπολογίζονται λιγότερο σύμφωνα με έναν παράγοντα μείωσης $\gamma = 0.95$. Το contour plot του συγκεκριμένου προβλήματος φαίνεται στην εικόνα 3.9.



Σχήμα 3.9: Το contour του δισδιάστατου προβλήματος που ορίζεται από τις 3.3 και 3.4

Δοκιμάσαμε τον MCEM αλγόριθμο για διαφορετικές αρχικές λύσεις με $\delta = 0.99, \max T = 100$, συλλέγοντας 50 τροχιές σε κάθε επανάληψη (για 300 επαναλήψεις) και παρατηρήσαμε τη σύγκλισή του. Στη συνέχεια, και για κάθε αρχική λύση, εφαρμόσαμε έναν αλγόριθμο MCEM με τις ίδιες παραμέτρους, με χρήση των 2 καλύτερων τροχιών. Τα αποτελέσματα για διάφορες αρχικές λύσεις φαίνονται στην εικόνα 3.10. Γίνεται πλέον ξεκάθαρο πως η χρήση των καλύτερων τροχιών επιταχύνει σημαντικά τον αλγόριθμο, ακόμη και σε περιβάλλοντα με εξαιρετικά υψηλά επίπεδα θορύβου, κάτι που δεν γίνεται αντιληπτό από τις καμπύλες μάθησης της εικόνας 3.6, αφού ο μεγάλος θόρυβος επηρεάζει την αποτίμηση της πολιτικής.



Σχήμα 3.10: Αριστερά: ο αλγόριθμος MCEM για διάφορες αρχικές λύσεις. Δεξιά: ο αλγόριθμος MCEM με χρήση των 2 καλύτερων τροχιών για τις ίδιες αρχικές λύσεις

3.2 Εξομοίωση αιώρησης ελικοπτέρου

Το πρόβλημα της αιώρησης ελικοπτέρου είναι ένα μη γραμμικό, υψηλής διάστασης και στοχαστικό πρόβλημα ελέγχου (Leishman, 2006; Bagnell and Schneider, 2001; Ng et al., 2004a,b). Σε αυτό το πείραμα χρησιμοποιούμε έναν εξομοιωτή ελικοπτέρου, ο οποίος παρέχεται από τους Pieter Abbeel, Adam Coates και Andrew Y. Ng του Πανεπιστημίου του Stanford και ο οποίος αποτελεί μια από τις κατηγορίες του Ετήσιου Διαγωνισμού Ενισχυτικής Μάθησης. Ο χώρος καταστάσεων εδώ έχει εννέα

διαστάσεις και περιλαμβάνει τη θέση του ελικοπτέρου (x, y, z), τη διεύθυνσή του (roll ϕ, pitch θ, yaw ω) και την ταχύτητά του σε κάθε διεύθυνση ($\dot{x}$, $\dot{y}$, $\dot{z}$). Ο χώρος των ενεργειών έχει τέσσερις διαστάσεις και αποτελείται από:

- u_1: Τον έλεγχο της μπρος - πίσω κυκλικής κλίσης (longitudinal cyclic pitch control)
- u_2: Τον έλεγχο της αριστερά - δεξιά κυκλικής κλίσης (latitudinal cyclic pitch control)
- u_3: Τον έλεγχο της κλίσης του ρότορα της κύριας έλικας
- u_4: Τον έλεγχο της κλίσης του ρότορα της ουραίας έλικας

Το πρόβλημα είναι η αιώρηση του ελικοπτέρου, αν κάθε στιγμή δίνεται ένα διάνυσμα λάθους εννέα διαστάσεων, το οποίο περιέχει την απόσταση καθεμιάς από τις παραμέτρους από ένα σημείο - στόχο στο χώρο των παραμέτρων. Ο ελεγκτής που χρησιμοποιείται για την επιτυχή αιώρηση του ελικοπτέρου είναι ο ακόλουθος:

$$u_1 = -w_1 x_{error} - w_2 \dot{x}_{error} - w_3 \theta_{error} + w_4 \tag{3.5}$$
$$u_2 = -w_5 y_{error} - w_6 \dot{y}_{error} - w_7 \phi_{error} + w_8 \tag{3.6}$$
$$u_3 = w_9 z_{error} - w_{10} \dot{z}_{error} + w_{11} \tag{3.7}$$
$$u_4 = -w_{12} \omega_{error} \tag{3.8}$$

Τα βάρη w_4, w_8 και w_{11} είναι σταθεροί όροι οι οποίοι χρησιμεύουν για το trimming του ελικοπτέρου. Σκοπός μας είναι να υπολογίσουμε το βέλτιστο σετ βαρών W^* μέσω μάθησης, χωρίς να γνωρίζουμε το μοντέλο της διεργασίας, όταν μας δίνεται μια αρχική λύση (ένα αρχικό διάνυσμα W_0).

Αρχικά, αποικοδομούμε το πρόβλημα σε τέσσερα υποπροβλήματα· αντί να προσπαθήσουμε να βρούμε τον καθολικό βέλτιστο ελεγκτή του προβλήματος, εφαρμόζουμε ξεχωριστά έναν αλγόριθμο ενισχυτικής μάθησης σε καθ' έναν από τους υπο-ελεγκτές των σχέσεων 3.5 - 3.8. Οι αλγόριθμοι χρησιμοποιούν τα ίδια δεδομένα από τις ίδιες τροχιές του συστήματος και υπολογίζουν τις νέες παραμέτρους τους ταυτόχρονα στο ίδιο βήμα. Η ανταμοιβή που συλλέγει το σύστημα σε κάθε χρονική στιγμή και διαμοιράζεται στους τέσσερις αλγορίθμους είναι ένα διάνυσμα της μορφής:

$$r_1(t) = exp(-x_{error}^2) + exp(-\dot{x}_{error}^2) + exp(-\theta_{error}^2) \tag{3.9}$$
$$r_2(t) = exp(-y_{error}^2) + exp(-\dot{y}_{error}^2) + exp(-\phi_{error}^2) \tag{3.10}$$
$$r_3(t) = exp(-z_{error}^2) + exp(-\dot{z}_{error}^2) \tag{3.11}$$
$$r_4(t) = exp(-\omega_{error}^2) \tag{3.12}$$

Αν οι τιμές του διανύσματος κατάστασης του ελικοπτέρου ξεπεράσουν κάποια προκαθορισμένα όρια, τότε το ελικόπτερο συντρίβεται. Συνοπτικά ο αλγόριθμος μάθησης για το πρόβλημα της αιώρησης ελικοπτέρου φαίνεται στον πίνακα 3.1.

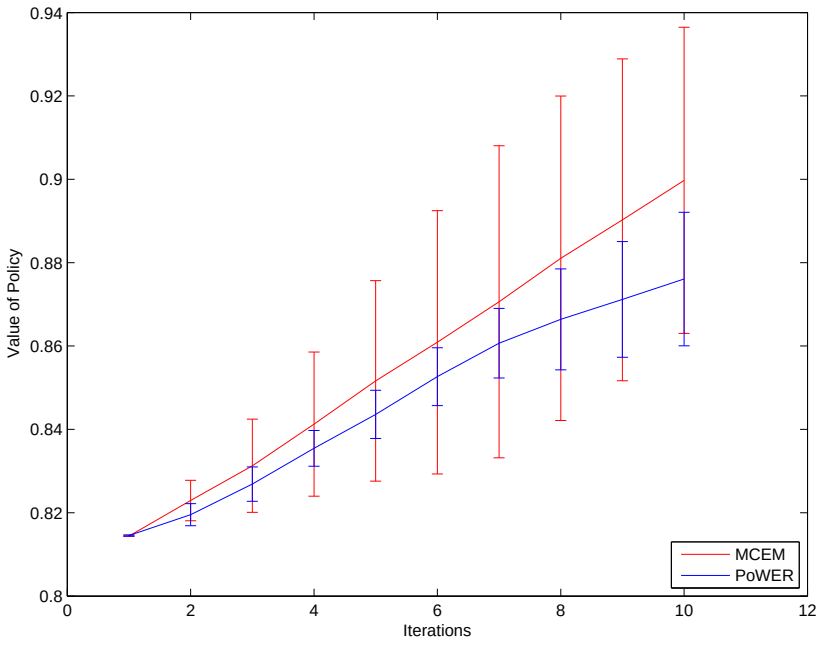
Δοκιμάζοντας τον PoWER και τον MCEM με διάφορες ρυθμίσεις των παραμέτρων τους στο πείραμα αυτό, διαπιστώσαμε πως και οι δύο αλγόριθμοι απέτυχαν να δώσουν κάποια ικανοποιητική λύση.

Πίνακας 3.1: Οι επιμέρους αλγόριθμοι μάθησης για το πρόβλημα της αιώρησης ελικοπτέρου

Ελεγκτής	Κατάσταση	Βάρη	Ανταμοιβή
Longitudinal	$x, \dot{x}, \theta$	w_1, w_2, w_3	$exp(-x^2_{error}) + exp(-\dot{x}^2_{error}) + exp(-\theta^2_{error})$
Latitudinal	$y, \dot{y}, \phi$	w_5, w_6, w_7	$exp(-y^2_{error}) + exp(-\dot{y}^2_{error}) + exp(-\phi^2_{error})$
Main Rotor	$z, \dot{z}$	w_9, w_{10}	$exp(-z^2_{error}) + exp(-\dot{z}^2_{error})$
Tail Rotor	ω	w_{12}	$exp(-\omega^2_{error})$

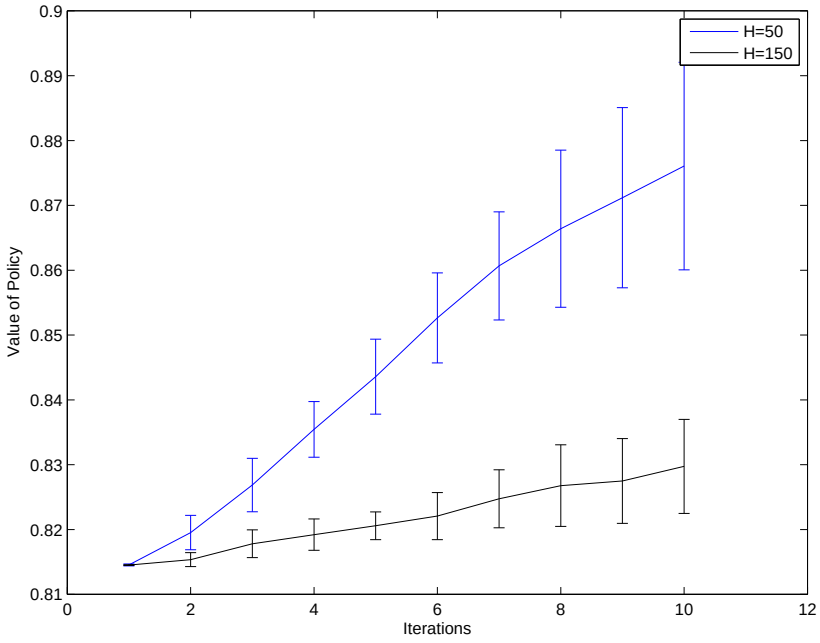
Για το λόγο αυτό χρησιμοποιήσαμε τις μεθόδους επιτάχυνσης του προηγούμενου κεφαλαίου και στους δύο αλγορίθμους. Για τον PoWER επιλέξαμε μήκος τροχιάς $H = 50$ (μετά από αρκετές δοκιμές) και για τον MCEM $\delta = 0.99$ και $\gamma = 0.95$ και μέγιστο επιτρεπόμενο μήκος τροχιάς $Tmax = 600$ βήματα. Κάθε επεισόδιο συνέχιζε έως τα 6000 βήματα, τα οποία αντιστοιχούν σε 10 λεπτά πτήσης του ελικοπτέρου σε πραγματικό χρόνο. Σε κάθε επανάληψη του αλγορίθμου χρησιμοποιήσαμε 100 επεισόδια και σε κάθε επανάληψη της αναζήτησης επί γραμμής 10 επεισόδια. Από τα 100 επεισόδια σε κάθε επανάληψη κρατούσαμε μόνο τα 5 καλύτερα, τα οποία χρησιμοποιούσαμε για τον υπολογισμό των νέων παραμέτρων. Οι ανταμοιβές συλλέγονταν σύμφωνα με τους κανόνες του διαγωνισμού ενισχυτικής μάθησης, δηλαδή δεν αξιολογούμε κάθε νέο σετ παραμέτρων με νέες τροχιές, αλλά ο ελεγκτής αξιολογείται κατά τη διάρκεια της μάθησης για όλα τα βήματα του επεισοδίου (6000). Αν έχουμε πτώση του ελικοπτέρου, τότε δίνεται στο σύστημα πολύ μεγάλη αρνητική ανταμοιβή για τα υπόλοιπα βήματα έως το τέλος του επεισοδίου.

Η απόδοση των δύο αλγορίθμων είναι συγκρίσιμη και φαίνεται στην εικόνα 3.11.



Σχήμα 3.11: Οι αλγόριθμοι MCEM και PoWER με αναζήτηση επί γραμμής και χρήση των καλύτερων τροχιών στο πρόβλημα της αιώρησης ελικοπτέρου

Στην εικόνα 3.12 φαίνεται η απόδοση του PoWER για $H = 150$, πράγμα που αποδεικνύει την ευαισθησία του αλγορίθμου στην επιλογή του μήκος του επεισοδίου. Από την άλλη πλευρά ο MCEM ξεπερνά αυτό το σκόπελο με σχετική ευκολία, χρησιμοποιώντας μεταβλητά μήκη τροχιών.



Σχήμα 3.12: Ο αλγόριθμος PoWER με αναζήτηση επί γραμμής και χρήση των καλύτερων τροχιών, για διαφορετικά μήκη τροχιάς H

3.3 RobBa (Robot Balancing)

Ένα μεγάλο μέρος της έρευνας για την δημιουργία σύγχρονων και αποδοτικών ρομποτικών ελεγκτών, έχει στραφεί σε δίτροχα, αυτο-ισορροπούμενα ρομπότ. Σήμερα υπάρχει πληθώρα τέτοιων κατασκευών, οι οποίες εξυπηρετούν εμπορικούς σκοπούς, όπως το γνωστό Segway® (Nguyen et al., 2004), ερευνητικούς σκοπούς (Grasser et al., 2002), αλλά και ψυχαγωγικούς σκοπούς ερασιτεχνών (Anderson, 2003).

Το τελευταίο πείραμα περιλαμβάνει ένα ισορροπούμενο δίτροχο ρομπότ, το οποίο ονομάζεται RobBa (RobotBalancing) και είναι μια ιδιοκατασκευή, η οποία φαίνεται στην εικόνα 3.13. Το RobBa χρησιμοποιεί διαφορική οδήγηση και είναι σχεδιασμένο έτσι ώστε να είναι ένα συμπαγές και χαμηλού κόστους αυτο-ισορροπούμενο ρομπότ. Αποτελείται από ένα πλαίσιο αλουμινίου, το οποίο συγκρατεί τα ακόλουθα μέρη:

→ Δύο 12 Vdc, 152 RpM spur geared κινητήρες

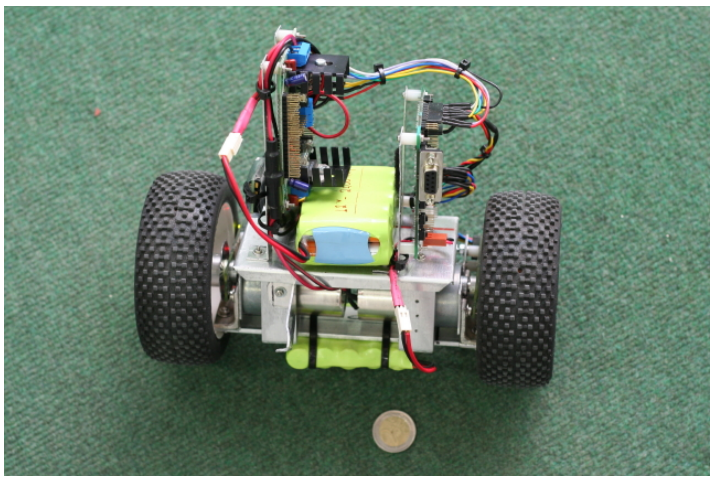
→ Έναν ooPIC microcontroller με το R carrier board

→ Δύο 64 παλμών ανά περιστροφή οδόμετρα, ένα για κάθε τροχό

→ Ένα επιταχυνσιόμετρο 2 αξόνων (ADXL203) και ένα γυροσκόπιο ενός άξονα (ADXRS300), και τα δύο της εταιρείας Analog Devices

→ Δύο πηγές ρεύματος, μια 12V 2700mAh επαναφορτιζόμενη μπαταρία για τους κινητήρες και μια 6V 2700mAh επαναφορτιζόμενη μπαταρία για τα ηλεκτρονικά υποσυστήματα

Μια από τις σημαντικότερες παραμέτρους που έπρεπε να ληφθούν υπ' όψιν κατά τη σχεδίαση του οχήματος ήταν η επίτευξη μικρών διαστάσεων και η στιβαρή κατασκευή, ικανή να αντεπεξέλθει σε τυχόν καταπονήσεις του οχήματος κατά την περίοδο μάθησης. Η RobBa χρησιμοποιεί δύο τροχούς διαμέτρου $12cm$, έχει μήκος $12cm$, πλάτος $24cm$, ύψος $21cm$ και ζυγίζει $2Kgr$.



Σχήμα 3.13: Το ρομπότ RobBa

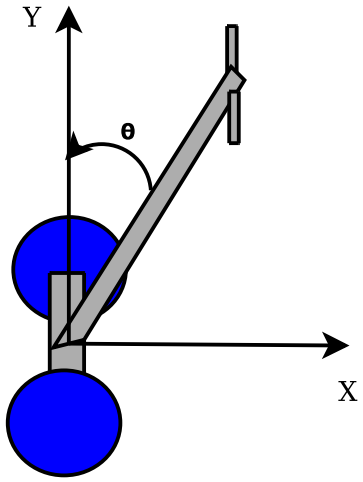
Στο πείραμά μας εκκινούμε το ρομπότ από μηδενική κάθετη γωνία (συν κάποιο θόρυβο) και δίνουμε στιγμιαία μεγάλη ροπή και στους δύο κινητήρες του ρομπότ, που έχει σαν αποτέλεσμα την απώλεια ισορροπίας του ρομπότ. Σκοπός μας είναι το ρομπότ να επανακάμψει στην αρχική του θέση όσο πιο γρήγορα γίνεται και να παραμείνει σε ισορροπία. Ο χώρος καταστάσεων εδώ έχει τέσσερις διαστάσεις και περιλαμβάνει την κάθετη γωνία x_1, τη γωνιακή ταχύτητα x_2, τη γραμμική ταχύτητα x_3 και τη μετατόπιση στον οριζόντιο άξονα x_4, οι οποίες φαίνονται καλύτερα και στην εικόνα 3.14. Αποδεικνύεται πως για τον πλήρη έλεγχο ενός τέτοιου οχήματος αρκεί ένας απλός PD ελεγκτής (Grasser et al., 2002), ο οποίος στην περίπτωσή μας είναι:

$$u_t = \boldsymbol{\theta}^T \boldsymbol{x}. \tag{3.13}$$

όπου $\boldsymbol{x} = [x_1, x_2, x_3, x_4]$ και $\boldsymbol{\theta} \in \mathbb{R}^4$. Σε κάθε χρονική στιγμή δίνουμε αρνητική ανταμοιβή σε κάθε γωνία διαφορετική από 0° και σε κάθε γραμμική ταχύτητα στον οριζόντιο άξονα του ρομπότ, χρησιμοποιώντας το ακόλουθο σχήμα για τις ανταμοιβές:

$$r(t) = exp(-x_1^2(t)) + exp(-x_3^2(t)) \tag{3.14}$$

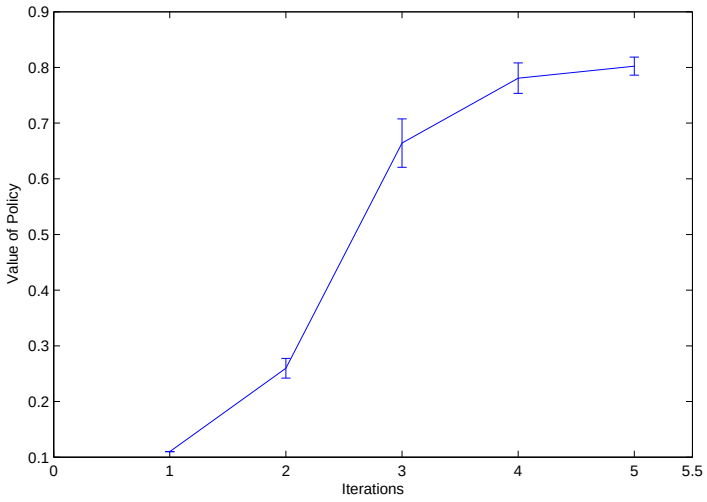
Για να έχουμε μια καλή εκτίμηση για τη γωνία του ρομπότ, μια και είναι γνωστό πως τα επιταχυνσιόμετρα και τα γυροσκόπια αντιμετωπίζουν συγκεκριμένα προβλήματα (Braunl, 2008) και παρέχουν



Σχήμα 3.14: Ένα δίτροχο αυτο-ισορροπούμενο όχημα στο χώρο

μετρήσεις που μπορεί να απέχουν σημαντικά από τις πραγματικές, υπολογίζαμε την εκτίμηση του συστήματος για την πραγματική γωνία, συγχωνεύοντας την εκτίμηση του επιταχυνσιόμετρου και την εκτίμηση του γυροσκόπιου σε ένα απλό φίλτρο Kalman (Kalman, 1960; Sorenson, 1985).

Σαν αρχικές παραμέτρους χρησιμοποιήσαμε λύσεις από αλγορίθμους που χρησιμοποιούν το μοντέλο του συστήματος και απευθύνονται σε παρόμοια προβλήματα (Kim et al., 2005), τις οποίες ρυθμίσαμε χειροκίνητα για να ανταποκρίνονται στα δεδομένα του συστήματός μας. Στο πείραμα με το πραγματικό ρομπότ χρησιμοποιήσαμε μόνο τον αλγόριθμο MCEM. Σε κάθε επανάληψη του αλγορίθμου συλλέγουμε 20 δειγματικές τροχιές, χρησιμοποιούμε $\gamma = 0.95$ και $\delta = 0.99$ και Impotrance Sampling. Μετά τη φάση της μάθησης εκτιμούμε την αξία κάθε ντετερμινιστικής πολιτικής που προέκυψε σε κάθε επανάληψη του αλγορίθμου, δοκιμάζοντάς την σε πέντε τροχιές στο ρομπότ, με μήκος επεισοδίου 200 βήματα. Η καμπύλη μάθησης φαίνεται στην εικόνα 3.15. Σημειώνεται πως η χρήση Importance Sampling επιτάχυνε σημαντικά τον αλγόριθμο, ώστε με λίγες μόνο επαναλήψεις του MCEM επέρχεται η επιθυμητή συμπεριφορά του συστήματος. Στο τέλος της μάθησης το ρομπότ μπορούσε να επανέρχεται γρήγορα στη θέση μετά από την επίδραση της αρχικής διαταραχής (αλλά και μετά από παρόμοιες χειροκίνητες διαταραχές) και να παραμένει στη θέση αυτή σε ισορροπία.



Σχήμα 3.15: Η καμπύλη μάθησης του RobBa με χρήση του αλγορίθμου MCEM και Importance Sampling

Κεφάλαιο 4

Συμπεράσματα και Μελλοντικές Επεκτάσεις

Στο κεφάλαιο αυτό παρατίθενται η συμβολή της παρούσας μεταπτυχιακής εργασίας στο ερευνητικό πεδίο της ρομποτικής μάθησης, καθώς και οι πιθανές μελλοντικές επεκτάσεις της εργασίας.

4.1 Συνεισφορά

Η κύρια συνεισφορά της εργασίας είναι η λεπτομερέστερη μελέτη της συμπεριφοράς των αλγορίθμων ενισχυτικής μάθησης μέσω πιθανοτικού συμπερασμού, η συμβολή στην ανάπτυξη ενός νέου αλγορίθμου στο πεδίο αυτό (Monte Carlo Expectation Maximization for Reinforcement Learning) και η ανάπτυξη και επαλήθευση μεθόδων επιτάχυνσης της συγκεκριμένης κατηγορίας αλγορίθμων.

Τα κύρια προτερήματα του αλγορίθμου MCEM είναι:

- Η πολύ απλή υλοποίησή του, σε σχέση με άλλους αλγορίθμους που απαιτούν πολύπλοκες μεθόδους (π.χ. παλινδρομήσεις, classifiers κ.ά).
- Ελάχιστη χειροκίνητη παραμετροποίηση, αφού το μόνο που απαιτείται είναι οι αρχικές τιμές των παραμέτρων για τους όρους εξερεύνησης ε και όχι η εύρεση και η ρύθμιση διαφόρων ρυθμών μάθησης.
- Η εύρωστη συμπεριφορά σε περιβάλλοντα με μεγάλη στοχαστικότητα.
- Η χαμηλή υπολογιστική του πολυπλοκότητα. Το μόνο που απαιτεί ο αλγόριθμος είναι μια σειρά από προσθέσεις και πολλαπλασιασμούς με πολυπλοκότητα $O(mT)$ (όπου m το πλήθος των τροχιών που συλλέγουμε σε κάθε επανάληψη και T το μήκος τους), η οποία είναι εν γένει μικρότερη σε σχέση με αλγορίθμους που χρησιμοποιούν classifiers (Dimitrakakis and Lagoudakis, 2008; Rexakis and Lagoudakis, 2008) ή προσπαθούν να κατασκευάσουν το μοντέλο του συστήματος (Nguyen-Tuong and Peters, 2008).

Από την άλλη πλευρά, οι σημαντικότερες αδυναμίες του αλγορίθμου MCEM είναι:

- Είναι ευαίσθητος σε τοπικά βέλτιστα, οπότε απαιτείται καλή αρχικοποίησή του.

- Επειδή χρησιμοποιεί τον αλγόριθμο EM δεν ακολουθεί την κατεύθυνση του πραγματικού διανύσματος κλίσης στο χώρο των παραμέτρων, αλλά αρκείται στην εύρεση μιας απλά καλύτερης λύσης στο επόμενο βήμα. Αυτό έχει σαν αποτέλεσμα να ακολουθεί μια "τεθλασμένη" πορεία προς το βέλτιστο, στο χώρο των παραμέτρων.
- Παρουσιάζει μεγάλες απαιτήσεις σε μνήμη. Αν για παράδειγμα συλλέγουμε 100 τροχιές σε κάθε επανάληψη του αλγορίθμου, έχουμε 10 παραμέτρους και μήκος επεισοδίου $H = 100$, τότε πρέπει να διατηρούμε στη μνήμη $100 \times 10 \times H = 100000$ όρους εξερεύνησης ε, οι οποίοι θα χρησιμεύσουν για την εύρεση νέων παραμέτρων.

Οι δύο κύριες μέθοδοι επιτάχυνσης αυτής της κατηγορίας αλγορίθμων που αναπτύξαμε, είδαμε πως επιταχύνουν σημαντικά τους δύο αλγορίθμους (MCEM και PoWER), χωρίς να επηρεάζουν τη σύγκλισή τους. Από την άλλη πλευρά, η μέθοδος αναζήτησης επί γραμμής απαιτεί πολλές τροχιές για την εκτίμηση της ποιότητας κάθε λύσης σε περιβάλλον με μεγάλη στοχαστικότητα και η μέθοδος χρήσης των καλύτερων τροχιών μπορεί να επηρεάσει τη σύγκλιση του αλγορίθμου, αν χρησιμοποιηθούν πολύ λίγες τροχιές. Σε κάθε περίπτωση, μερικοί εμπειρικοί κανόνες για τη χρήση των μεθόδων επιτάχυνσης είναι οι ακόλουθοι:

- Χρήση πάντα των $p\%$ καλύτερων τροχιών, αλλά με προσεκτική επιλογή του p
- Αν το περιβάλλον δεν είναι πολύ στοχαστικό, χρήση και της μεθόδου αναζήτησης επί γραμμής

Τα περιεχόμενα της παρούσας μεταπτυχιακής εργασίας συνέβαλλαν στη δημοσίευση των ακόλουθων εργασιών:

1. Nikos Vlassis, Kontes Georgios and Savas Piperidis, "Reinforcement Learning of Robot Control via Probabilistic Inference", 1st Hellenic Robotics Conference (HEROC), February 23-24, Athens, 2009
2. Nikos Vlassis, Marc Toussaint, Georgios Kontes and Savas Piperidis, "Learning Model - free Robot Control by a Monte Carlo EM Algorithm", Autonomous Robots, Special issue on robot learning (accepted)

4.2 Μελλοντικές Επεκτάσεις

Ένας από τους άμεσους στόχους για το μέλλον είναι η επέκταση των αλγορίθμων που περιγράφονται στην παρούσα εργασία στο πλαίσιο της κατανεμημένης ενισχυτικής μάθησης. Σήμερα υπάρχει μια πλειάδα αλγορίθμων που προσπαθούν να επιλύσουν το πρόβλημα της ενισχυτικής μάθησης ενός συνόλου από πράκτορες (Guestrin et al., 2002; Bowling and Veloso, 2002; Hu and Wellman, 2003), κανείς όμως από τους οποίους δεν επεκτείνεται σε συνεχείς χώρους καταστάσεων και ενεργειών. Η φυσική επέκταση της εργασίας είναι η ενσωμάτωση των συγκεκριμένων αλγορίθμων στις μεθοδολογίες αυτές και η επέκτασή τους σε προβλήματα πολυπρακτορικού ελέγχου σε πραγματικά ρομποτικά συστήματα.

Επιπλέον, επειδή όπως είδαμε ο αλγόριθμος MCEM δεν ακολουθεί το πραγματικό διάνυσμα κλίσης, στόχος είναι η συνένωσή του με έναν αλγόριθμο πολιτικής κλίσης (ιδανικά τον eNAC), έτσι

ώστε η κατεύθυνση του διανύσματος κλίσης να δίνεται από τον eNAC και το μέγεθος του βήματος προς την κατεύθυνση αυτή από τον MCEM.

4.3 Συμπεράσματα

Η παρούσα εργασία μελέτησε την συμπεριφορά και την απόδοση των αλγορίθμων PoWER και MCEM και πρότεινε δύο μεθόδους επιτάχυνσής τους. Η ταχύτητα και η ποιότητα της σύγκλισης των αλγορίθμων που προέκυψαν παρουσιάστηκε σε απλά συνθετικά προβλήματα και σε ένα πιο απαιτητικό (αιώρηση ελικοπτέρου). Τέλος, οι δυνατότητες του αλγορίθμου MCEM αποδείχτηκαν και στη μάθηση ενός πραγματικού ρομποτικού συστήματος

Βιβλιογραφία

Anderson, D. (2003). nBot balancing robot. *URL: http://www. geology. smu. edu/dpa-www/robo/nbot*.

Bagnell, J. and Schneider, J. (2001). Autonomous helicopter control using reinforcement learning policy search methods. In *IEEE International Conference on Robotics and Automation, 2001. Proceedings 2001 ICRA*, volume 2.

Barto, A., Sutton, R., and Anderson, C. (1988). Neuronlike adaptive elements that can solve difficult learning control problems.

Bazaraa, M., Sherali, H., and Shetty, C. (2006). *Nonlinear programming: theory and algorithms*. Wiley-interscience.

Bellman, R. and Dreyfus, S. (1962). *Applied dynamic programming*. Princeton University Press.

Bertsekas, D. (1987). *Dynamic programming: deterministic and stochastic models*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA.

Bertsekas, D., Homer, M., Logan, D., and Patek, S. (1995). Nonlinear programming. *Athena scientific*.

Bertsekas, D. and Tsitsiklis, J. (1996). *Neuro-dynamic programming*. Athena Scientific.

Bowling, M. and Veloso, M. (2002). Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250.

Braunl, T. (2008). *Embedded robotics: mobile robot design and applications with embedded systems*. Springer-Verlag New York Inc.

Chang, H., Fu, M., Hu, J., and Marcus, S. (2007). *Simulation-based algorithms for Markov decision processes*. Springer Berlin.

Cooper, G. (1988). A method for using belief networks as influence diagrams. In *Workshop on Uncertainty in Artificial Intelligence*, pages 55–63.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38.

Dimitrakakis, C. and Lagoudakis, M. (2008). Rollout sampling approximate policy iteration. *Machine Learning*, 72(3):157–171.

Duflo, M. and Wilson, S. (1997). *Random iterative models*. Springer Verlag.

Ernst, D., Geurts, P., and Wehenkel, L. (2006). Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(1):503.

Fidelman, P. and Stone, P. (2004). Learning ball acquisition on a physical robot. In *2004 International Symposium on Robotics and Automation (ISRA)*.

Gaskett, C., Wettergreen, D., and Zelinsky, A. (1999). Q-learning in continuous state and action spaces. *Lecture notes in computer science*, pages 417–428.

Glynn, P. (1987). Likelilood ratio gradient estimation: an overview. In *Proceedings of the 19th conference on Winter simulation*, pages 366–375. ACM New York, NY, USA.

Grasser, F., D' Arrigo, A., Colombi, S., and Rufer, A. (2002). JOE: a mobile, inverted pendulum. *IEEE Transactions on industrial electronics*, 49(1):107–114.

Guestrin, C. G., Lagoudakis, M., and Parr, R. (2002). Coordinated Reinforcement Learning. In *Proceedings of the ICML-2002 The Nineteenth International Conference on Machine Learning*, pages 227–234.

Hoffman, M., Doucet, A., De Freitas, N., Jasra, A., Platt, J., Koller, D., Singer, Y., and Roweis, S. (2008). Bayesian policy learning with trans-dimensional mcmc. *Advances in neural information processing systems*, 20.

Howard, R. (1960). *Dynamic programming and Markov process*. MIT press.

Hu, J. and Wellman, M. (2003). Nash Q-learning for general-sum stochastic games. *The Journal of Machine Learning Research*, 4:1039–1069.

Kaelbling, L., Littman, M., and Moore, A. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence*, 4(1):237–285.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45.

Kim, Y., Kim, S., and Kwak, Y. (2005). Dynamic analysis of a nonholonomic two-wheeled inverted pendulum robot. *Journal of Intelligent and Robotic Systems*, 44(1):25–46.

Kober, J. and Peters, J. (2008). Policy search for motor primitives in robotics. *Submitted to NIPS08*.

Kohl, N. and Stone, P. (2004). Policy gradient reinforcement learning for fast quadrupedal locomotion. In *2004 IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04*, volume 3.

Lagoudakis, M. and Parr, R. (2003). Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149.

Lawrence, G., Cowan, N., and Russell, S. (2003). Efficient gradient estimation for motor control learning. In *Uncertainty in Artificial Intelligence*, pages 354–36. Morgan Kaufmann.

Lazaric, A., Restelli, M., and Bonarini, A. (2007). Reinforcement learning in continuous action spaces through sequential monte carlo methods. *Advances in neural information processing systems*.

Leishman, J. (2006). *Principles of helicopter aerodynamics*. Cambridge Univ Pr.

Mannor, S., Rubinstein, R., and Gat, Y. (2003). The Cross Entropy method for Fast Policy Search. In *International Conference on Machine Learning*, pages 512–519. Morgan Kaufmann.

Neal, R. and Hinton, G. (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. *Learning in graphical models*, 89:355–368.

Ng, A., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., and Liang, E. (2004a). Inverted autonomous helicopter flight via reinforcement learning. In *International Symposium on Experimental Robotics*, pages 406–415. Citeseer.

Ng, A., Kim, H., Jordan, M., Sastry, S., and Ballianda, S. (2004b). Autonomous helicopter flight via reinforcement learning. *Advances in Neural Information Processing Systems*, 16.

Nguyen, H., Morrell, J., Mullens, K., Burmeister, A., Miles, S., Farrington, N., Thomas, K., and Gage, D. (2004). Segway robotic mobility platform. *SPIE Mobile Robots XVII*.

Nguyen-Tuong, D. and Peters, J. (2008). Local Gaussian process regression for real-time model-based robot control. pages 380–385.

Pazis, J. and Lagoudakis, M. (2009). Binary action search for learning continuous-action control policies. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM New York, NY, USA.

Peters, J. and Schaal, S. (2006). Policy gradient methods for robotics. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2219–2225.

Peters, J. and Schaal, S. (2008). Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190.

Puterman, M. (1994). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, Inc. New York, NY, USA.

Rexakis, I. and Lagoudakis, M. (2008). Classifier-based policy representation. In *8th European Workshop on Reinforcement Learning*.

Rubinstein, R. and Kroese, D. (2004). *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning*. Springer-Verlag New York Inc.

Ruckstie, T., Felder, M., and Schmidhuber, J. (2008). State-dependent exploration for policy gradient methods. In *European Conference on Machine Learning (ECML) and Principles and Practice of Knowledge Discovery in Databases*, pages 234–249. Springer.

Santamaria, J., Sutton, R., and Ram, A. (1997). Experiments with reinforcement learning in problems with continuous state and action spaces. *Adaptive behavior*, 6(2):163.

Sorenson, H. (1985). *Kalman filtering: theory and application*. IEEE press New York.

Spall, J. (2005). *Introduction to stochastic search and optimization: estimation, simulation, and control*. Wiley-Interscience.

Sutton, R., McAllester, D., Singh, S., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12:1057–1063.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning : An Introduction*. MIT press, Cambridge, Massachusetts.

Tanner, M. (1990). A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, pages 699–704.

Toussaint, M. and Storkey, A. (2006). Probabilistic inference for solving discrete and continuous state Markov Decision Processes. In *Proceedings of the 23rd international conference on Machine learning*, pages 945–952. ACM New York, NY, USA.

Vlassis, N. and Toussaint, M. (2009). Model-free Reinforcement Learning as Mixture Learning.

Vlassis, N., Toussaint, M., Kontes, G., and Piperidis, S. (2009). Learning Model - free Robot Control by a Monte Carlo EM Algorithm. *Autonomous Robots, Special issue on robot learning*.

Watkins, C. (1989). *Learning from delayed rewards*. University of Cambridge.

Williams, R. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256.