ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Τεχνικές Βελτιστοποίησης:

Ελαχιστοποίηση συναρτήσεων πολλών μεταβλητών χωρίς περιορισμούς με χρήση παραγώγων

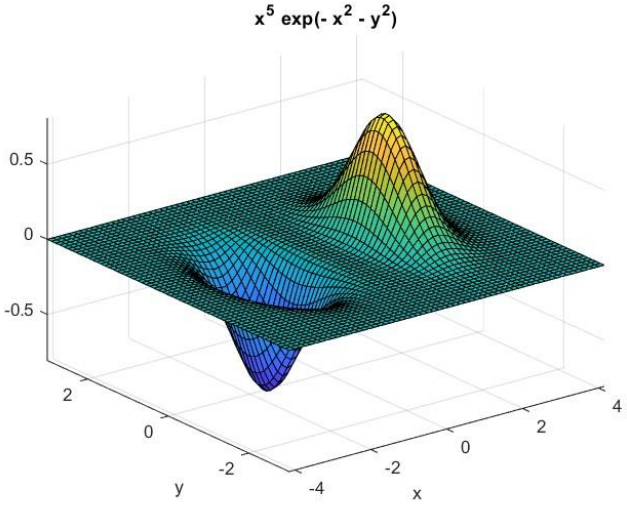
Αφροδίτη Λάτσκου

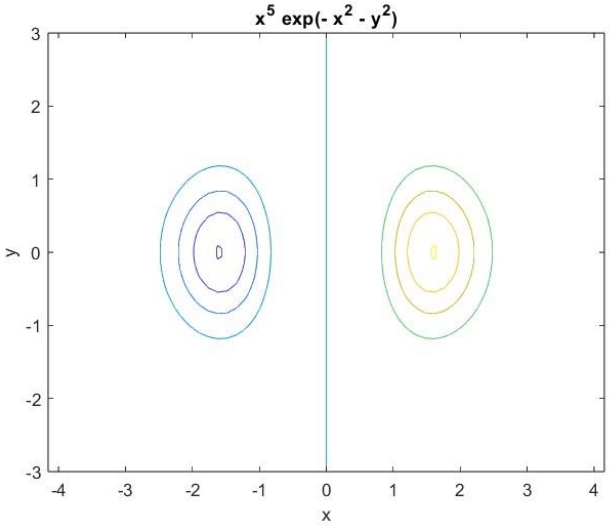
Νοέμβριος 2024

Σύνοψη εργασίας

Σκοπός μας ήταν να ελαχιστοποιήσουμε την συνάρτηση:

$$f(x,y) = x^5 e^{-x^2-y^2}$$





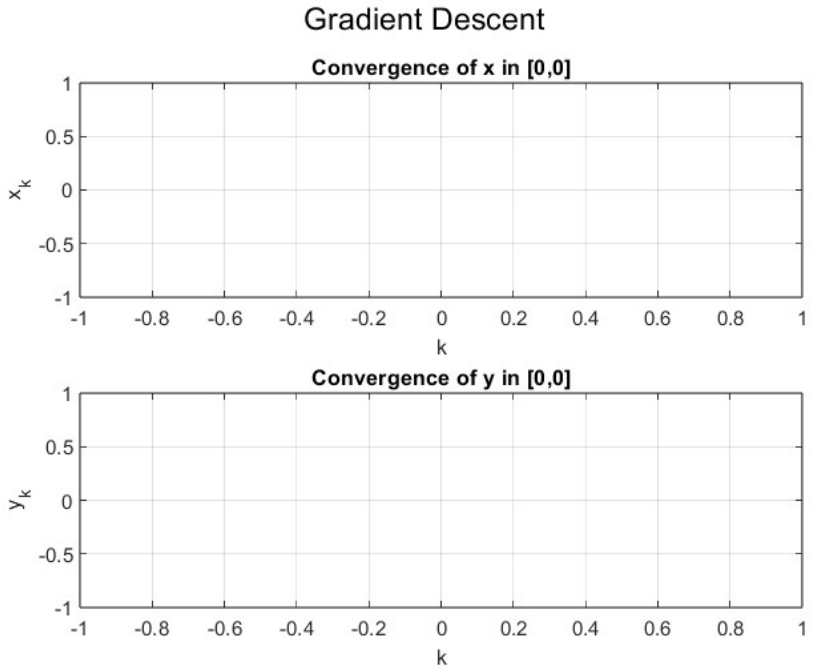
Από τα γραφήματα μας παρατηρούμε πως η συνάρτηση είναι κοίλη σε ένα σημείο και κυρτή σε ένα άλλο, καθώς και ότι έχει παραπάνω από ένα τοπικό μέγιστο και ελάχιστο. Αυτό μας προμηνύει ήδη πως υπάρχει κίνδυνος εγκλωβισμού των αλγορίθμων σε κάποιο τοπικό ελάχιστο που δεν είναι και το ολικό. Από τις ισοϋψείς καμπύλες ιδιαίτερα παίρνουμε μια καλύτερη ιδέα του γύρω από ποιες τιμές θα έχουμε τελικά το τοπικό ελάχιστο.

Τα σημεία που μας ενδιαφέρουν είναι τα [0, 0], [-1, 1], [1, -1]. Παρατηρούμε πως αυτά βρίσκονται πάνω στο επίπεδο 0 που έχει πολλά τοπικά ελάχιστα, στο downward slope προς το ολικό ελάχιστο και στο upward slope προς το ολικό μέγιστο αντίστοιχα. Αυτό ήδη μας δίνει hint για την συμπεριφορά των αλγορίθμων σε κάθε ένα από αυτά τα σημεία, και σε πιο τοπικό ελάχιστο είναι πιθανότερο να καταλήξουν. Ήδη περιμένουμε δηλαδή τα σημεία [0, 0] και [1, -1] να μη φτάσουν ποτέ στο ολικό ελάχιστο.

1. Μέθοδος της Μέγιστης Καθόδου

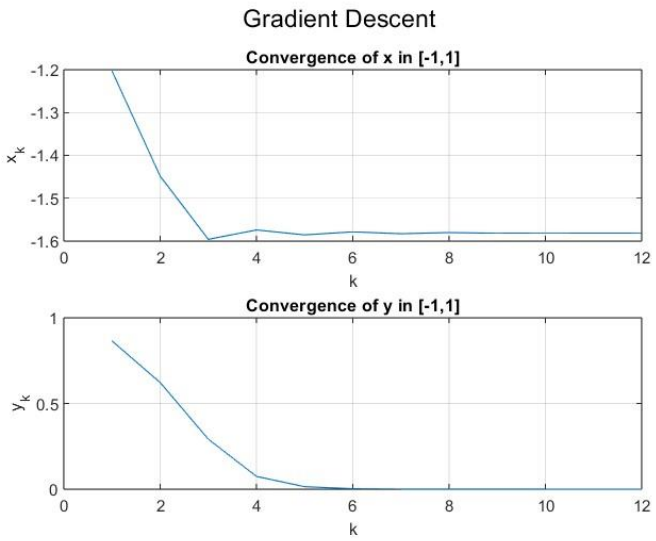
Η μέθοδος αυτή, σε αντίθεση με τις επόμενες που θα δούμε, χρησιμοποιεί μόνο το gradient της συνάρτησης για να βρει το μονοπάτι που θα ελαχιστοποιήσει την f. Θα δούμε πως αλλάζει το ελάχιστο στο οποίο σταθεροποιείται ο αλγόριθμος με βάση το σημείο εκκίνησης για [0,0]. [-1,1] και [1,-1] με βήμα 0.5.

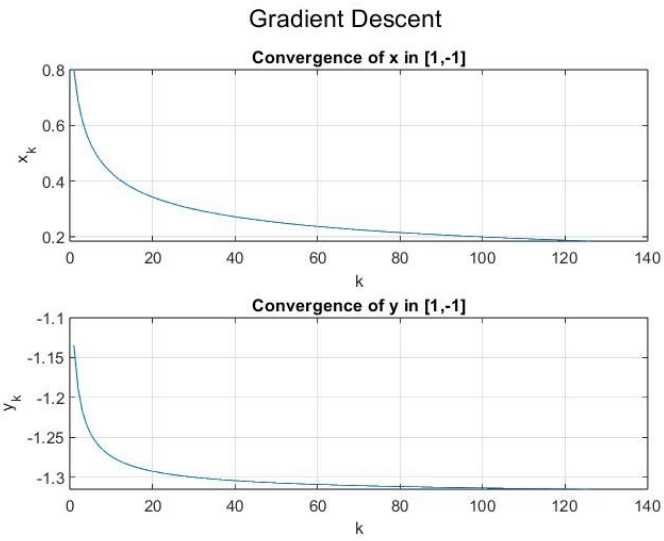
Για αρχή, όσον αφορά το [0,0], τόσο σε αυτή την περίπτωση, όσο και σε κάθε άλλη, το αποτέλεσμα που παίρνουμε είναι το παρακάτω.



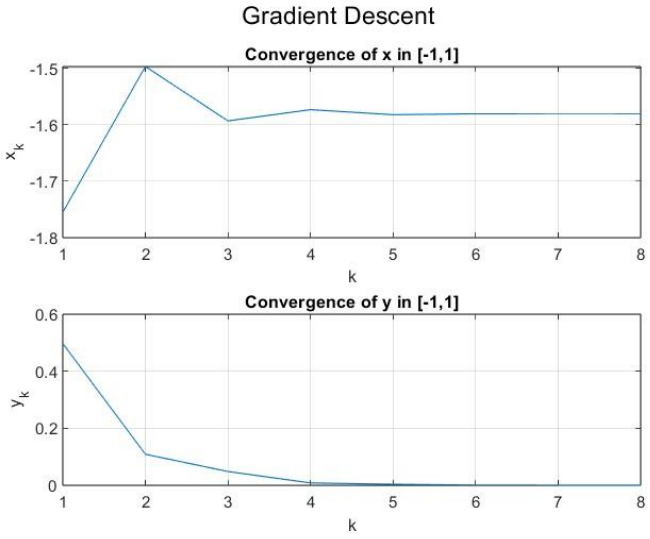
Ανατρέχοντας στις γραφικές παραστάσεις, βλέπουμε πως πράγματι στο [0,0] υπάρχει τοπικό ελάχιστο, επομένως ο αλγόριθμος εγκλωβίζεται σε αυτό το σημείο και δεν κάνει κανένα βήμα προς το ελάχιστο που ψάχνουμε. Για αυτόν τον λόγο θα παραλείπεται από δω και πέρα, εφόσον το ίδιο ισχύει και στις υπόλοιπες μεθόδους.

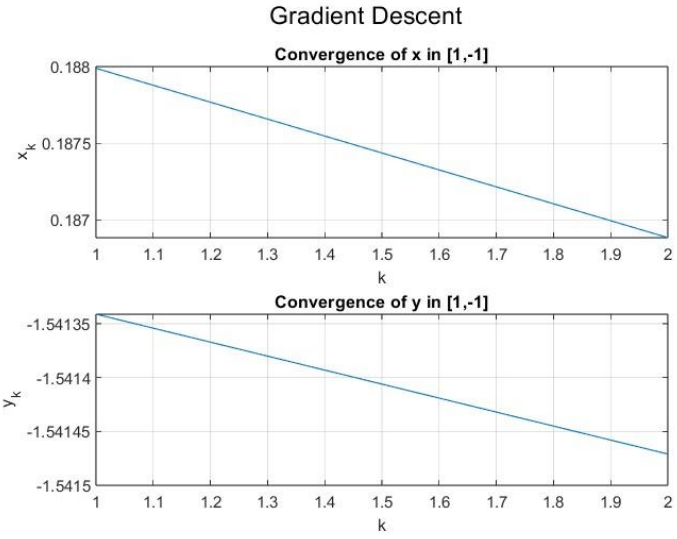
Παρακάτω παραθέτονται τα αποτελέσματα για τα αρχικά σημεία [-1, 1] και [1, -1]. Όπως φαίνεται, χρησιμοποιώντας αρχικό σημείο το [-1, 1], φτάνουμε πράγματι στο επιθυμητό ελάχιστο που βρίσκεται στο σημείο [-1.5810, 0] με τιμή -0.8112. Ωστόσο, όταν το αρχικό σημείο είναι στο [-1, 1], τότε ο αλγόριθμος εγκλωβίζεται στο τοπικό ελάχιστο 0, και μάλιστα μετά από πολλές επαναλήψεις.





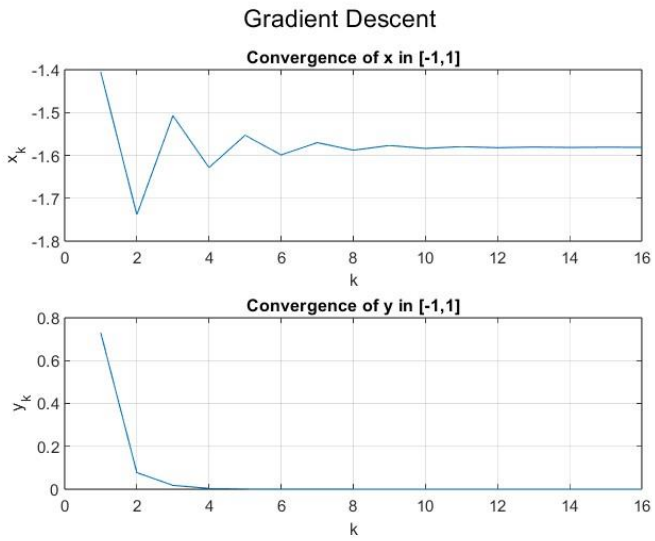
Στη συνέχεια, αντί για ένα αυθαίρετο step θεωρούμε ένα γ τέτοιο ώστε να ελαχιστοποιεί την $f(x_k + \gamma_k d_k)$. Παίρνουμε έτσι, τα εξής αποτελέσματα:

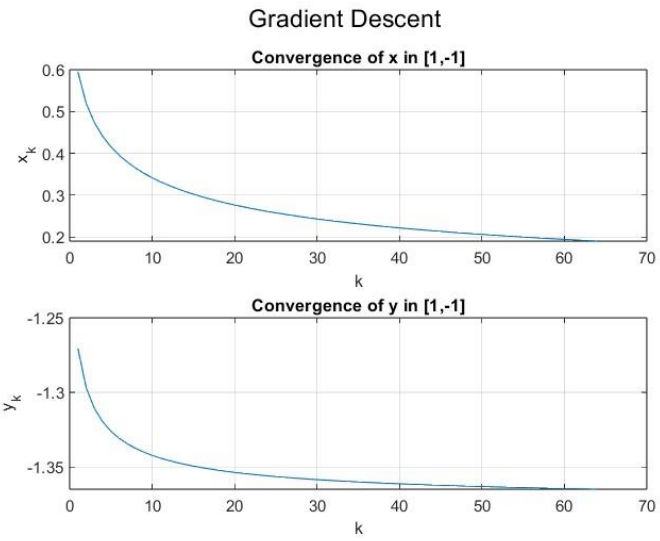




Όσον αφορά τις τελικές τιμές για το σημείο και την τιμή ελαχίστου, τα αποτελέσματα είναι σχεδόν πανομοιότυπα. Ωστόσο, παρατηρούμε πως τα iterations που χρειάστηκαν είναι αρκετά λιγότερα, και ειδικά όταν πρόκειται για το σημείο [-1, 1]. Ο τρόπος σύγκλισης επίσης αλλάζει, πάλι ειδικά στην δεύτερη περίπτωση.

Δοκιμάζοντας επίσης την επιλογή του step με τον κανόνα Armijo:

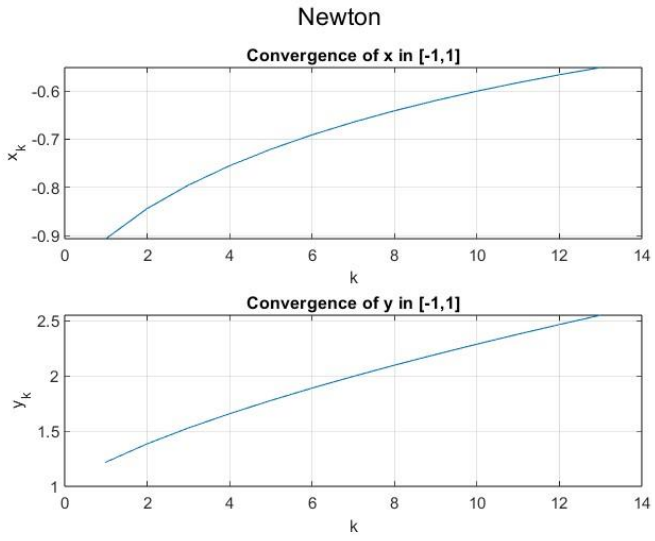


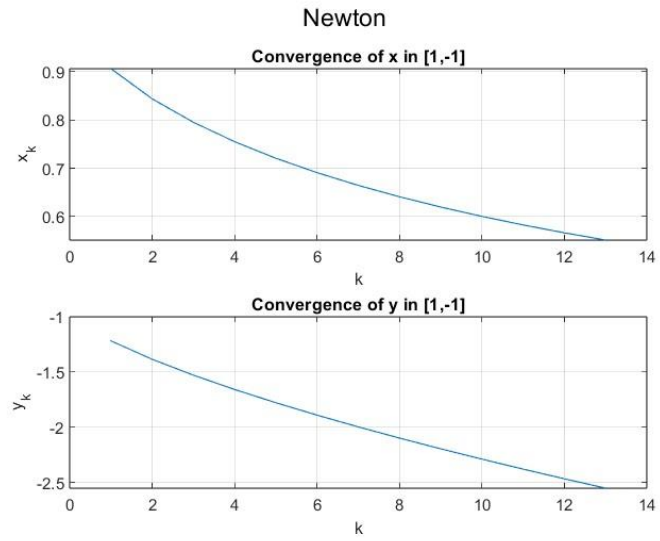


Ξανά, τα τελικά αποτελέσματα παραμένουν τα ίδια και αλλάζει ο τρόπος σύγκλισης. Ωστόσο αυτή την φορά έχουμε λίγο μεγαλύτερους χρόνους αλλά χωρίς κάποια ενίσχυση στην ακρίβεια.

2. Μέθοδος Newton

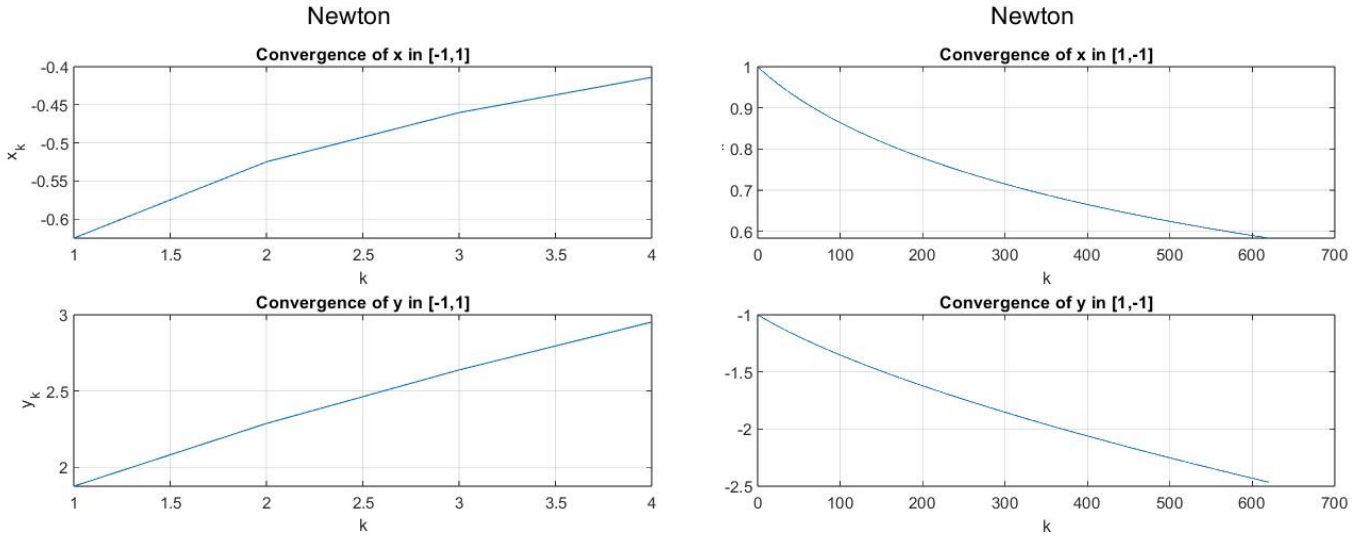
Αυτή η μέθοδος χρησιμοποιεί, πέρα από το gradient, και τον Εσσιανό πίνακα, και συγκεκριμένα τον αντίστροφο του, πολλαπλασιασμένο με το gradient σε κάθε σημείο ώστε να βρει την επόμενη βέλτιστη κατεύθυνση.





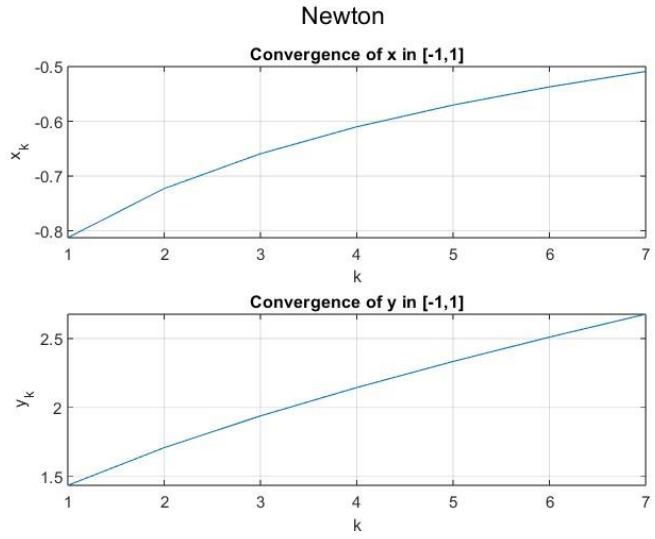
Όπως παρατηρούμε, με την μέθοδο Newton, ο αλγόριθμος εγκλωβίζεται και για τα δύο σημεία στο τοπικό ελάχιστο με τιμή 0. Όπως βλέπουμε από την γραφική παράσταση, σε αυτό το ελάχιστο υπάρχουν πολλά σημεία, κάτι που εξηγεί το ότι δεν συγκλίνουν στο ίδιο σημείο, αλλά συγκλίνουν στην ίδια ελάχιστη τιμή. Εν τούτοις, η σύγκριση με το gradient descent ως προς τα iterations δε θα ήταν πολύ δίκαιη, εφόσον συγκλίνουν σε τελείως άλλα σημεία. Γενικά, από αυτά μπορούμε να συμπεράνουμε πως πιθανότατα ο αλγόριθμος αυτός να μην είναι κατάλληλος για την επίλυση του συγκεκριμένου προβλήματος με την συγκεκριμένη συνάρτηση.

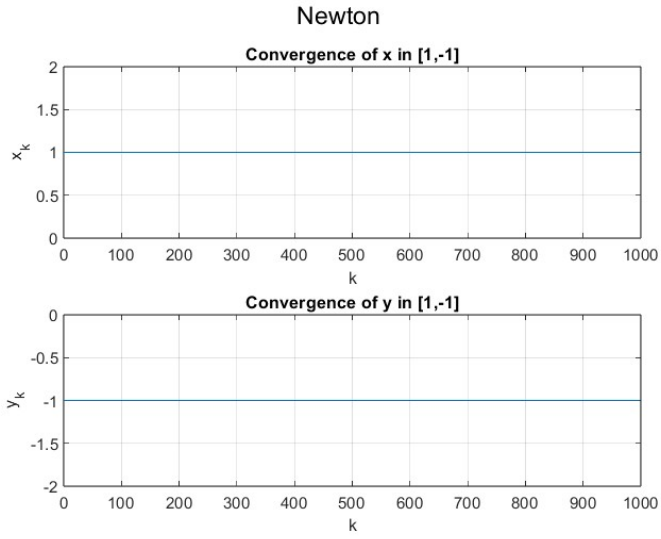
Εάν αντί για ένα αυθαίρετο step θεωρήσουμε ξανά ένα γ τέτοιο ώστε να ελαχιστοποιεί την $f(x_k + \gamma_k d_k)$:



Παρατηρούμε πως και πάλι άλλαξαν λίγο τα σημεία αλλά παραμένει ίδια η τιμή του τοπικού ελαχίστου στο οποίο εγκλωβίζεται ο αλγόριθμος. Τα iterations, σε σχέση με την προηγούμενη επιλογή step, για το αρχικό σημείο [-1, 1] μειώθηκαν ενώ για το [1, -1] αυξήθηκαν. Αυτό το γεγονός μπορεί να σχετίζεται με το πόσο κοντά είναι το κάθε σημείο στο σημείο όπου καταλήγει, ή και κάποιο λάθος στην υλοποίηση της συνάρτησης για την εύρεση του κατάλληλου γ.

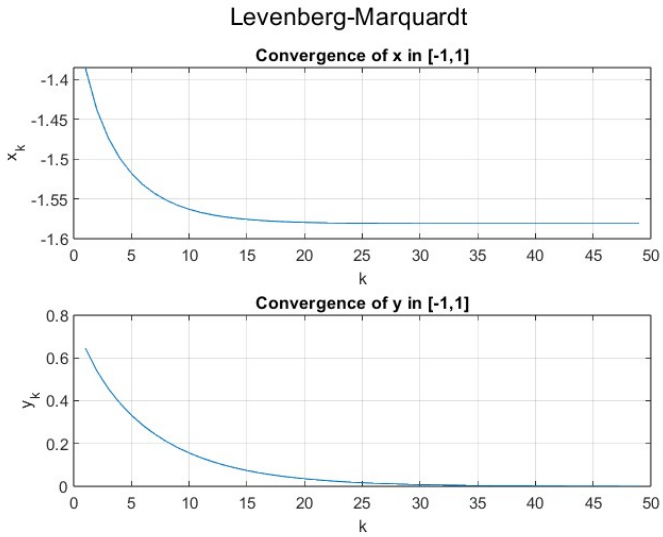
Δοκιμάζοντας και τον κανόνα Armijo στον συγκεκριμένο αλγόριθμο:

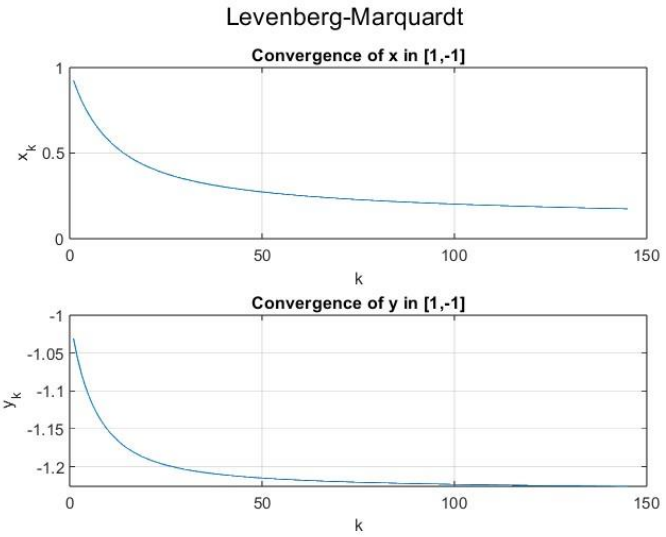




Βλέπουμε πως τα αποτελέσματα όσον αφορά το αρχικό σημείο [-1, 1] δεν άλλαξαν σημαντικά, ωστόσο αυξήθηκαν λίγο οι επαναλήψεις, ενώ το [1, -1] δεν καταφέρνει καν να συγκλίνει κάπου και μένει στο ίδιο σημείο, από το οποίο, εκτός αν υπάρχει λάθος στον κώδικα, θα μπορούσε να οφείλεται στο ότι ο κανόνας Armijo δεν είναι κατάλληλη επιλογή για τον ορισμό του step στην μέθοδο Newton.

3. Μέθοδος Levenberg-Marquardt

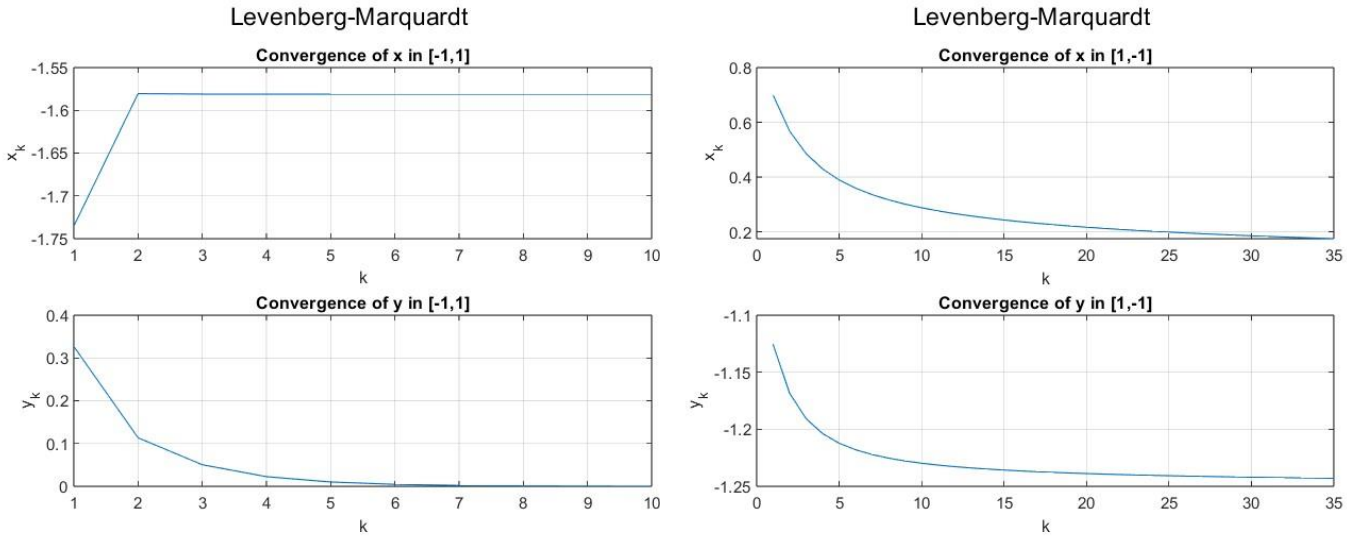




Η τελευταία μέθοδος πάει να γεφυρώσει το κενό ανάμεσα στη μέθοδο της μέγιστης καθόδου και στη μέθοδο Newton. Οι νέες τιμές που παίρνουν τα x, y κάθε φορά εξαρτώνται από έναν παράγοντα μ, ο οποίος και καθορίζει αν η μέθοδος θα δρα περισσότερο σαν την μια ή την άλλη εκ των μεθόδων που αναφέρθηκαν πιο πάνω.

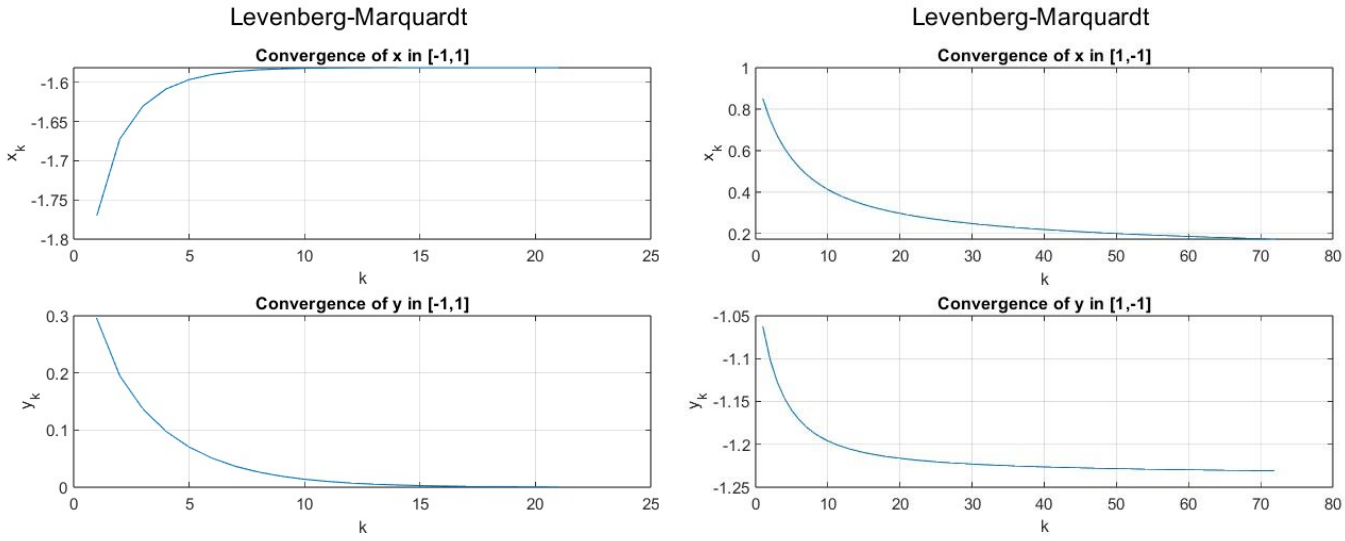
Πιο πάνω φαίνονται τα αποτελέσματα για step 0.5. Βλέπουμε πως για αρχικό σημείο [-1, 1] ο αλγόριθμος φτάνει όντως στο ολικό ελάχιστο -0.8112 στο σημείο [-1.5810, 0], σε αντίθεση με τον αλγόριθμο Newton, ωστόσο με αρκετά iterations, καθιστώντας το αργότερο από τον αλγόριθμο του gradient descent. Για το σημείο [1, -1] έχουμε πάλι εγκλωβισμό σε σημείο με τιμή τοπικού ελαχίστου 0.

Με step που δίνεται από την ελαχιστοποίηση του $f(x_k + \gamma_k d_k)$:



Παρατηρούμε πως η σύγκλιση γίνεται πολύ γρηγορότερα και για τα δύο σημεία και μάλιστα με εξαιρετική ακρίβεια, ενώ τα αποτελέσματα παραμένουν σχεδόν ίδια όσον αφορά τις τιμές.

Δοκιμάζοντας τον κανόνα Armijo:



Εδώ έχουμε περισσότερα iterations για τα ίδια αποτελέσματα, οπότε πιθανότατα να μην προτιμούσαμε αυτόν τον τρόπο επιλογής βήματος.

Γενικότερα, μπορούμε να πούμε πως ο αλγόριθμος της μέγιστης καθόδου υπερίσχυσε στη συγκεκριμένη περίπτωση και των δύο άλλων, εφόσον βρήκε το σωστό ελάχιστο για το σημείο που το καθιστούσε δυνατό αλλά και είναι πολύ πιο γρήγορος και απλός στην υλοποίηση. Φυσικά, η αποδοτικότητα των μεθόδων εξαρτάται πολύ από την ίδια την συνάρτηση που έχουν ως είσοδο, από το βήμα, το αρχικό σημείο (όπως άλλωστε αποδείχθηκε) αλλά και την βελτιστοποίηση του ίδιου του κώδικα. Επομένως, πιθανότατα με άλλα δεδομένα, οι ίδιοι αλγόριθμοι να έδιναν καλύτερα ή και χειρότερα αποτελέσματα.