Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Τριτη Εργαστηριακη Αναφορα
Νευροασαφης Έλεγχος

Αναστασία Χριστίνα Λίβα
03119029

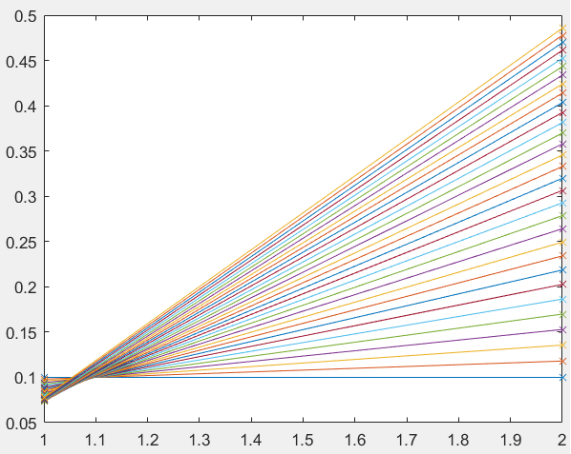
Περιεχόμενα

Μέρος Πρώτο

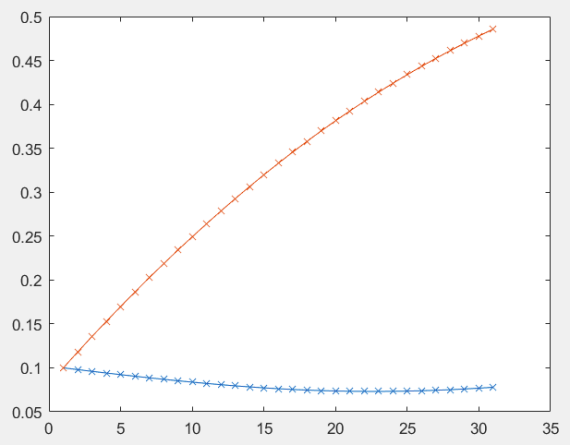
Θέμα 1

Σκοπός του θέματος αυτού είναι η αξιολόγηση της σύγκλισης της μεθόδου Gradient Descent κατά την επίλυση του προβλήματος βελτιστοποίησης για τη συνάρτηση $f(x) = x_1^2 + (x_2 - 1)^2 + (x_1 - x_2)^4$. Επιπλέον, πραγματοποιείται σύγκριση με τη μέθοδο Newton, προκειμένου να εξεταστούν οι διαφορές στη σύγκλιση των δύο μεθόδων.

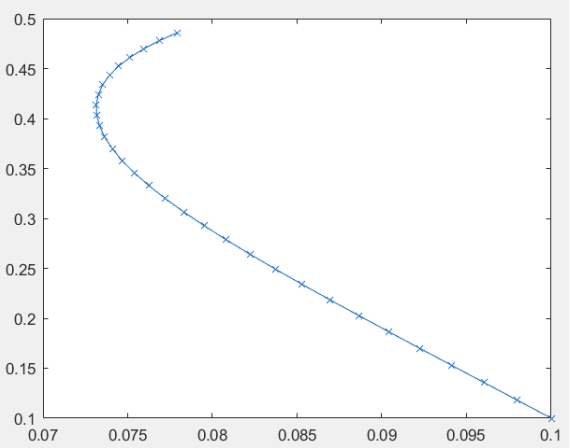
Οι μέθοδοι Gradient Descent και Newton υλοποιήθηκαν σε περιβάλλον MATLAB. Για την εξέταση της επίδρασης του μεγέθους βήματος στη σύγκλιση, εκτελέστηκε ο κώδικας για διάφορες τιμές του `step_size` οπότε και προέκυψαν τα εξής αποτελέσματα:



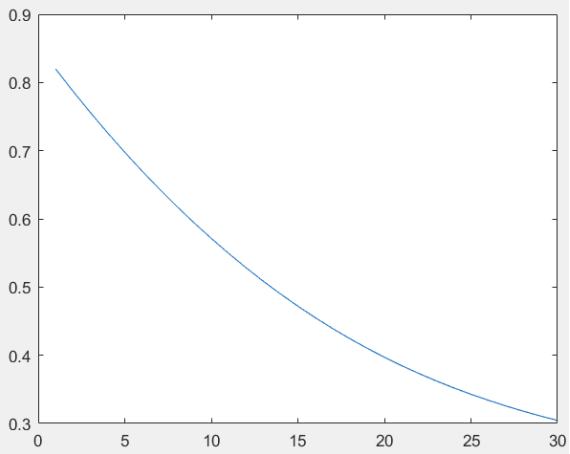
Σχήμα 1: Καμπύλη σύγκλισης x με τη μέθοδο Gradient Descent



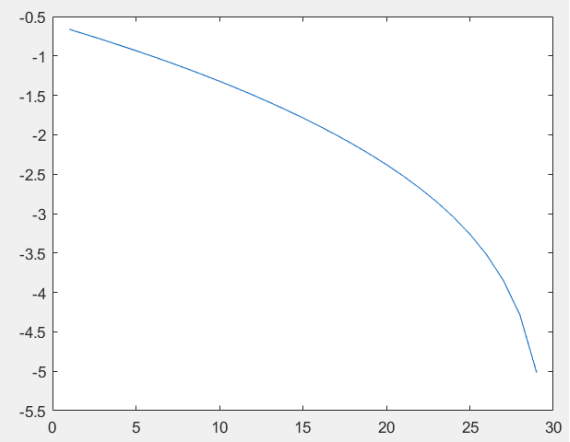
Σχήμα 2: x_1 και x_2



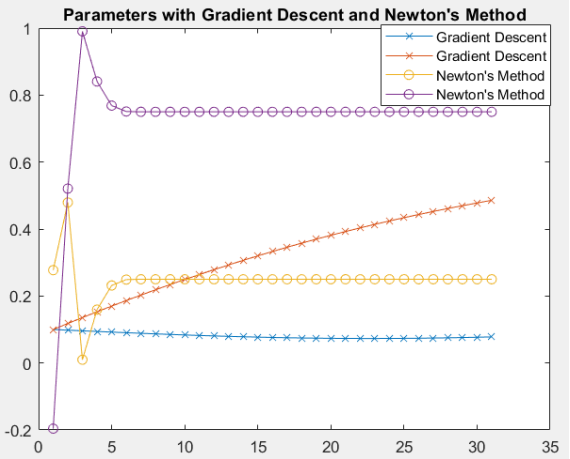
Σχήμα 3: Συνδυασμός των δύο plots



Σχήμα 4: Τιμή Συνάρτησης



Σχήμα 5: Λογάριθμος Συνάρτησης



Σχήμα 6: Σύγκριση μεταξύ Μεθόδου Gradient Descent και Μεθόδου Newton

Συμπεράσματα: Η σύγκριση ανάμεσα στη μέθοδο Gradient Descent και τη Μέθοδο Newton αποτυπώνει τις διαφορές στη σύγκλιση και την απόδοσή τους στην επίλυση της συγκεκριμένης συνάρτησης. Αναλυτικότερα:

Η Gradient Descent παρουσιάζει σύγκλιση του διανύσματος x προς το ελάχιστο της συνάρτησης, με την ταχύτητα σύγκλισης να εξαρτάται σημαντικά από το μέγεθος του βήματος. Η εξέλιξη της συνάρτησης και η διαδρομή των σημείων x παρουσιάζονται σε αντίστοιχα γραφήματα, ενώ το γράφημα ρυθμού σύγκλισης προσφέρει μια ποσοτική εκτίμηση της σύγκλισης.

Από την άλλη πλευρά, η Μέθοδος Newton επιδεικνύει γρήγορη σύγκλιση, εκμεταλλευόμενη την πληροφορία του Hessian για την ενημέρωση του διανύσματος x. Η μέθοδος Newton δεν εφαρμόζεται στην συνάρτηση κόστους αλλά στην παράγωγο της συνάρτησης κόστους, αφού βρίσκει ρίζες και

όχι (τοπικά) μέγιστα/ελάχιστα. Σε αυτό οφείλεται και το γεγονός ότι παρατηρείται περισσότερη γραμμικότητα στα αποτελέσματα της μεθόδου Newton, συγκριτικά με το Gradient Descent.

Θέμα 2

Στην άσκηση αυτή έχω τη συνάρτηση $f(x) = Ax_1^2 + \frac{1}{A}x_2^2$, όπου $A > 0$, και μελετώ την ταχύτητα σύγκλισης του αλγορίθμου gradient descent για διάφορες τιμές του A. Επιπλέον, υπολογίζω το Condition number του Hessian πίνακα με σκοπό να αξιολογήσω την επίδραση της τιμής του A στην απόδοση του αλγορίθμου.

Ανάλυση Αλγορίθμου Gradient Descent: Αρχικά, θέτω $a = 0.04$ και ALPHA $= 10$, και κατασκευάζω τον πίνακα A ως εξής:

$$A = \begin{bmatrix} 1 - a^2 \cdot \text{ALPHA} & 0 \\ 0 & 1 - 2 \cdot \frac{a}{\text{ALPHA}} \end{bmatrix}$$

Έπειτα, υπολογίζω Condition number του πίνακα A και μελετώ τη συμπεριφορά του κατά την εκτέλεση του αλγορίθμου gradient descent για 20 και 100 βήματα.

Αποτελέσματα και Συμπεράσματα: Ο πίνακας A δεν είναι καθολικά καλά συμπεριφερόμενος, καθώς το Condition number του δείχνει την ευαισθησία του αλγορίθμου στις αλλαγές του A. Συγκεκριμένα, παρατηρώ ότι με την αύξηση του A, το Condition number αυξάνεται, καθιστώντας τον αλγόριθμο πιο ευαίσθητο στις αρχικές συνθήκες.

Για να αντιμετωπίσω αυτό το πρόβλημα, χρησιμοποιώ τη μέθοδο της ορμής. Η εισαγωγή της ορμής βελτιώνει τη σύγκλιση του αλγορίθμου, χρησιμοποιώντας τις παραμέτρους $\alpha_1 = 0.1$ και $\alpha_2 = 0.1$. Η εκτέλεση του αλγορίθμου με ορμή παρατείνεται για 200 επαναλήψεις, και καταλήγει σε σύγκλιση των λύσεων κατά τη διάρκεια των επαναλήψεων.

Τελικά, η μέθοδος της ορμής αποδεικνύεται αποτελεσματική στη βελτίωση της σύγκλισης του αλγορίθμου gradient descent, ειδικά όταν βρίσκομαι αντιμέτωπη με προβλήματα που έχουν υψηλό Condition number.

```
>> ex3_2

A =

    0.9840         0
         0    0.9920

Condition number of the matrix A: 1.008130
After 20 steps:

ans =

    0.7243         0
         0    0.8516

After 100 steps:

ans =

    0.1993         0
         0    0.4479
```

Σχήμα 7: Αποτελέσματα για πίνακα A

```
Iteration 10: x = [0.059861, -0.320140]
Iteration 20: x = [0.004395, -0.255559]
Iteration 30: x = [0.000323, -0.204006]
Iteration 40: x = [0.000024, -0.162852]
Iteration 50: x = [0.000002, -0.130000]
Iteration 60: x = [0.000000, -0.103776]
Iteration 70: x = [0.000000, -0.082841]
Iteration 80: x = [0.000000, -0.066130]
Iteration 90: x = [0.000000, -0.052790]
Iteration 100: x = [0.000000, -0.042141]
Iteration 110: x = [0.000000, -0.033640]
Iteration 120: x = [0.000000, -0.026854]
Iteration 130: x = [0.000000, -0.021436]
Iteration 140: x = [0.000000, -0.017112]
Iteration 150: x = [0.000000, -0.013660]
Iteration 160: x = [0.000000, -0.010905]
Iteration 170: x = [0.000000, -0.008705]
Iteration 180: x = [0.000000, -0.006949]
Iteration 190: x = [0.000000, -0.005547]
Iteration 200: x = [0.000000, -0.004428]
```

Σχήμα 8: Σύγκλιση

Θέμα 3

Σκοπός του παρόντος εργαστηριακού θέματος είναι η ανάλυση του Condition Number του πίνακα Q, όπου Q ορίζεται ως το γινόμενο ενός τυχαίου πίνακα A με τον ανάστροφό του ($Q = AA^t$). Εξετάζω το πώς μεταβάλλεται το Condition Number του Q σε σχέση με τη διάσταση του πίνακα.

Υλοποίηση σε MATLAB: Ξεκινώ με τη δημιουργία τυχαίου πίνακα A για διάσταση $N = 4$:

```
rng(42);
N = 4;
A = rand(N, N);

Q = A*A';
eigenvalues_Q = eig(Q);

fprintf('N = 4  -> Condition Number:\n')
condQ_N4 = max(eigenvalues_Q) / min(eigenvalues_Q);
```

Στη συνέχεια, επαναλαμβάνω την ίδια διαδικασία για μεγαλύτερη διάσταση, $N = 20$:

```
N = 20;
A = rand(N, N);

Q = A*A';
eigenvalues_Q = eig(Q);

fprintf('N = 20 -> Condition Number:\n')
condQ_N20 = max(eigenvalues_Q) / min(eigenvalues_Q);
```

Αποτελέσματα: Αρχικά, για μικρή διάσταση $N = 4$, το Condition Number (condQ_N4) παραμένει σε λογικά επίπεδα, δείχνοντας ότι ο πίνακας Q είναι σχετικά καλά συνθηκικός. Ωστόσο, όσο αυξάνεται η διάσταση του πίνακα ($N = 20$), το Condition Number (condQ_N20) αυξάνεται σημαντικά. Αυτή η αύξηση υποδεικνύει ότι ο πίνακας Q γίνεται όλο και πιο ill-conditioned, καθιστώντας την επίλυση συστημάτων εξισώσεων που περιλαμβάνουν τον πίνακα Q πιθανώς πιο ευαίσθητη σε αριθμητικά σφάλματα.

Συνεπώς, η αύξηση της διάστασης του πίνακα επηρεάζει σημαντικά τη συνθήκη του Q, και οι εφαρμογές που απαιτούν τη χρήση του Q θα πρέπει να λαμβάνουν υπόψη τις επιπτώσεις αυτής της αυξανόμενης ευαισθησίας στην ακρίβεια των υπολογισμών τους.

```
>> ex3_3

ans =

    0.0673
    0.4453
    0.5497
    4.2870

N = 4   -> Condition Number:

condQ =

   63.7199

N = 20 -> Condition Number:

condQ =

   1.5914e+05
```

Σχήμα 9: Αποτελέσματα

Θέμα 4

Ερευνώ τη σύγκλιση της μεθόδου του gradient descent σε μία δοσμένη συνάρτηση. Η συνάρτηση που εξετάζω είναι η εξής:

$$f(x_1, x_2) = \max(x_1 + x_2, 0.9x_1 - 1.1x_2 + 1, -0.8x_1 + 1.2x_2 - 1, 2 - 1.1x_1 - 0.9x_2)$$

Η εφαρμογή της μεθόδου του gradient descent θα με βοηθήσει να κατανοήσμε τη συμπεριφορά της συνάρτησης αυτής καθώς και τον τρόπο σύγκλισής της.

Υλοποίηση: Χρησιμοποιώ τη μέθοδο του gradient descent. Αρχικά, ορίζω τη συνάρτηση f και δημιουργώ ένα πλέγμα σημείων στο επίπεδο x_1, x_2 ώστε οπτικοποιήσω τη συνάρτηση.

Στη συνέχεια, θέτω $learning_rate = 0.1$ και $iterations = 50$. Επιλέγω ένα αρχικό σημείο $x = [-3, -5]$ και εκτελώ τον αλγόριθμο gradient descent.

Η συμπεριφορά της μεθόδου εξετάζεται κατά τη διάρκεια των επαναλήψεων, καθώς και μεταβάλλοντας το μήκος του βήματος. Επιπλέον, ελέγχω τη σύγκλιση της μεθόδου όταν χρησιμοποιώ βήμα $\frac{1}{k}$.

Αποτελέσματα και Συμπεράσματα: Κατά την εκτέλεση της μεθόδου του gradient descent, παρατηρώντας την εξέλιξη του σημείου x κατά τις επαναλήψεις βλέπω πως στο τέλος έχω συμπεριφορά ταλάντωσης. Μπορώ να ελαττώσω το μέγεθος της ταλάντωσης ελαττώνοντας το step size. Παίρνοντας βήμα της $\frac{1}{k}$ βλέπω πως το πλάτος της ταλάντωσης έχει μειωθεί σημαντικά

```
>> ex3_4
-2.890000e+00
-4.910000e+00
-2.780000e+00
-4.820000e+00
-2.670000e+00
-4.730000e+00
-2.560000e+00
-4.640000e+00
-2.450000e+00
-4.550000e+00
-2.340000e+00
-4.460000e+00
-2.230000e+00
-4.370000e+00
-2.120000e+00
-4.280000e+00
-2.010000e+00
-4.190000e+00
-1.900000e+00
-4.100000e+00
-1.790000e+00
-4.010000e+00
-1.680000e+00
-3.920000e+00
-1.570000e+00
-3.830000e+00
-1.460000e+00



-3.740000e+00
-1.350000e+00
-3.650000e+00
-1.240000e+00
-3.560000e+00
-1.130000e+00
-3.470000e+00
-1.020000e+00
-3.380000e+00
-9.100000e-01
-3.290000e+00
-8.000000e-01
-3.200000e+00
-6.900000e-01
-3.110000e+00
-5.800000e-01
-3.020000e+00
-4.700000e-01
-2.930000e+00
-3.600000e-01
-2.840000e+00
-2.500000e-01
-2.750000e+00
-1.400000e-01
-2.660000e+00
-3.000000e-02
-2.570000e+00
8.000000e-02
```

```
-2.480000e+00
1.900000e-01
-2.390000e+00
3.000000e-01
-2.300000e+00
2.100000e-01
-2.190000e+00
3.200000e-01
-2.100000e+00
2.300000e-01
-1.990000e+00
3.400000e-01
-1.900000e+00
2.500000e-01
-1.790000e+00
3.600000e-01
-1.700000e+00
2.700000e-01
-1.590000e+00
3.800000e-01
-1.500000e+00
2.900000e-01
-1.390000e+00
4.000000e-01
-1.300000e+00
3.100000e-01
-1.190000e+00
4.200000e-01
```

```
-1.100000e+00
3.300000e-01
-9.900000e-01
4.400000e-01
-9.000000e-01
3.500000e-01
-7.900000e-01
4.600000e-01
-7.000000e-01
3.700000e-01
-5.900000e-01
4.800000e-01
-5.000000e-01
3.900000e-01
-3.900000e-01
5.000000e-01
-3.000000e-01
```

Θέμα 5

Εξετάζω την εφαρμογή της μεθόδου στοχαστικού gradient descent σε μια συγκεκριμένη συνάρτηση ορισμένη ως:

$$f(x_1, x_2) = \frac{1}{4}\sum_{n=1}^{4}[(x_1 - a_i)^2 + (x_2 - b_i)^2]$$

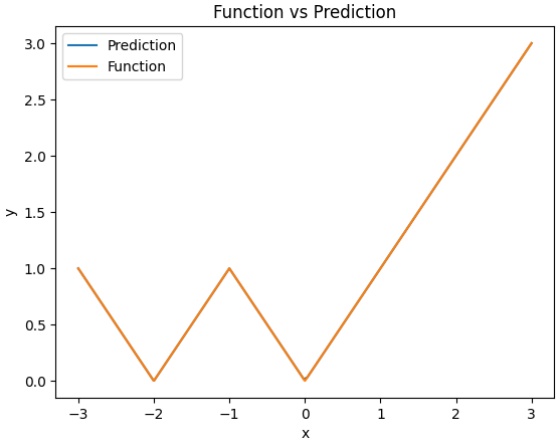
Υποθέτω ότι κατά την κάθε επανάληψη της μεθόδου, μπορώ να υπολογίσω μόνο ένα από τα τέσσερα $d_i = \frac{1}{2}[(x_1 - a_i), (x_2 - b_i)]^T$. Ο στόχος είναι να ελαχιστοποιήσω τη συνάρτηση f χρησιμοποιώντας τη μέθοδο στοχαστικού gradient descent.

Αποτελέσματα και Συμπεράσματα: Βλέπω ότι, παρά την τυχαιότητα που εισήγαγα στον υπολογισμό του gradient, η μέθοδος στοχαστικού gradient descent συγκλίνει αρκετά γρήγορα. Η συμπεριφορά του αλγορίθμου επηρεάζεται σημαντικά από τον ρυθμό μάθησης, με διάφορες τιμές να οδηγούν σε διαφορετικούς «τρόπους» σύγκλισης.

Τα αποτελέσματα που εμφανίζονται στον πίνακα 'points' αντικατοπτρίζουν τις ενδιάμεσες τιμές του x κατά τη διάρκεια των επαναλήψεων. Επιπλέον, η εκτύπωση των τιμών του x κατά την εξέλιξη του training είναι χρήσιμη για την παρακολούθηση της σύγκλισης του αλγορίθμου.

Μέρος Δεύτερο

Θέμα 1



Σχήμα 10: Αποτελέσματα

```
Epoch 0, Loss: 1.5449317693710327
Epoch 1000, Loss: 0.02209840714931488
Epoch 2000, Loss: 0.009211830794811249
Epoch 3000, Loss: 2.1005609596613795e-05
Epoch 4000, Loss: 3.1721850973553956e-05
```

```
Epoch 5000, Loss: 1.0683501159292064e-06
Epoch 6000, Loss: 3.3833043033837384e-08
Epoch 7000, Loss: 4.912600548578894e-09
Epoch 8000, Loss: 6.502441607381115e-08
Epoch 9000, Loss: 1.771915236759014e-08
```

Θέμα 2

Για τις αρχικές παραμέτρους με optimizer Adam έχω:

```
Epoch 1/10
1875/1875 [==============================] - 12s 6ms/step - loss: 0.5005
    - accuracy: 0.8244
Epoch 2/10
1875/1875 [==============================] - 13s 7ms/step - loss: 0.3742
    - accuracy: 0.8650
Epoch 3/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3354
    - accuracy: 0.8773
Epoch 4/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.3121
    - accuracy: 0.8841
Epoch 5/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2958
    - accuracy: 0.8902
Epoch 6/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2803
    - accuracy: 0.8957
Epoch 7/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2671
    - accuracy: 0.9006
Epoch 8/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2578
    - accuracy: 0.9040
Epoch 9/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2482
    - accuracy: 0.9068
Epoch 10/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2397
    - accuracy: 0.9099
```

Με optimizer SGD έχω:

```
Epoch 1/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.7478
    - accuracy: 0.7557
Epoch 2/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.5131
    - accuracy: 0.8251
```

```
Epoch 3/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.4685
    - accuracy: 0.8391
Epoch 4/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.4441
    - accuracy: 0.8467
Epoch 5/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.4258
    - accuracy: 0.8526
Epoch 6/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.4124
    - accuracy: 0.8582
Epoch 7/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.4007
    - accuracy: 0.8604
Epoch 8/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3909
    - accuracy: 0.8637
Epoch 9/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.3817
    - accuracy: 0.8671
Epoch 10/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3736
    - accuracy: 0.8704
```

Με optimizer RMSprop έχω:

```
Epoch 1/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.5043
    - accuracy: 0.8216
Epoch 2/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3763
    - accuracy: 0.8644
Epoch 3/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.3460
    - accuracy: 0.8768
Epoch 4/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3272
    - accuracy: 0.8828
Epoch 5/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.3123
    - accuracy: 0.8887
Epoch 6/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3030
    - accuracy: 0.8929
Epoch 7/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2938
    - accuracy: 0.8964
Epoch 8/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2870
    - accuracy: 0.8988
Epoch 9/10
```

```
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2817
    - accuracy: 0.9021
Epoch 10/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2760
    - accuracy: 0.9062
```

Με optimizer Adagrad έχω:

```
    Epoch 1/10
1875/1875 [==============================] - 7s 4ms/step - loss: 1.0981
    - accuracy: 0.6654
Epoch 2/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.7318
    - accuracy: 0.7624
Epoch 3/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.6528
    - accuracy: 0.7891
Epoch 4/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.6111
    - accuracy: 0.8016
Epoch 5/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.5834
    - accuracy: 0.8100
Epoch 6/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.5632
    - accuracy: 0.8161
Epoch 7/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.5474
    - accuracy: 0.8206
Epoch 8/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.5346
    - accuracy: 0.8240
Epoch 9/10
1875/1875 [==============================] - 7s 3ms/step - loss: 0.5241
    - accuracy: 0.8266
Epoch 10/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.5152
    - accuracy: 0.8301
```

Με optimizer Anadelta έχω:

```
Epoch 1/10
1875/1875 [==============================] - 9s 4ms/step - loss: 2.1867
    - accuracy: 0.1447
Epoch 2/10
1875/1875 [==============================] - 8s 4ms/step - loss: 1.8599
    - accuracy: 0.4051
Epoch 3/10
1875/1875 [==============================] - 7s 4ms/step - loss: 1.6091
    - accuracy: 0.5575
Epoch 4/10
```

```
1875/1875 [==============================] - 8s 4ms/step - loss: 1.4214
    - accuracy: 0.6085
Epoch 5/10
1875/1875 [==============================] - 9s 5ms/step - loss: 1.2824
    - accuracy: 0.6363
Epoch 6/10
1875/1875 [==============================] - 7s 4ms/step - loss: 1.1777
    - accuracy: 0.6556
Epoch 7/10
1875/1875 [==============================] - 8s 4ms/step - loss: 1.0967
    - accuracy: 0.6700
Epoch 8/10
1875/1875 [==============================] - 9s 5ms/step - loss: 1.0331
    - accuracy: 0.6817
Epoch 9/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.9820
    - accuracy: 0.6924
Epoch 10/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.9402
    - accuracy: 0.7030
```

Με optimizer AdamW έχω:

```
Epoch 1/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.4984
    - accuracy: 0.8260
Epoch 2/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3769
    - accuracy: 0.8640
Epoch 3/10
1875/1875 [==============================] - 12s 6ms/step - loss: 0.3389
    - accuracy: 0.8770
Epoch 4/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.3141
    - accuracy: 0.8837
Epoch 5/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2998
    - accuracy: 0.8912
Epoch 6/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2826
    - accuracy: 0.8963
Epoch 7/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2714
    - accuracy: 0.9000
Epoch 8/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2586
    - accuracy: 0.9044
Epoch 9/10
1875/1875 [==============================] - 9s 5ms/step - loss: 0.2509
    - accuracy: 0.9065
Epoch 10/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2401
```

```
    - accuracy: 0.9106
```

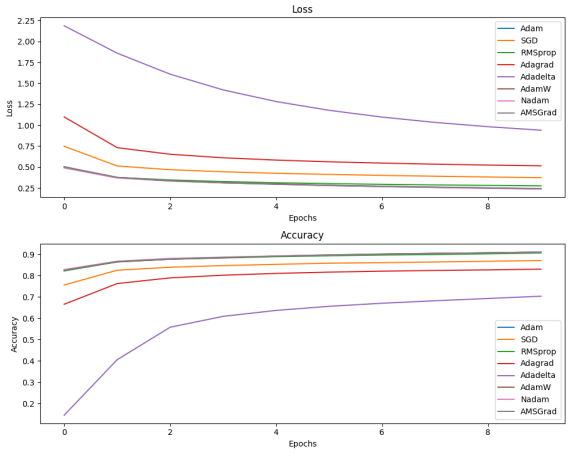
Με optimizer Nadam έχω:

```
Epoch 1/10
1875/1875 [==============================] - 9s 4ms/step - loss: 0.4894
    - accuracy: 0.8282
Epoch 2/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.3699
    - accuracy: 0.8672
Epoch 3/10
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3328
    - accuracy: 0.8804
Epoch 4/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.3113
    - accuracy: 0.8862
Epoch 5/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2940
    - accuracy: 0.8920
Epoch 6/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2791
    - accuracy: 0.8965
Epoch 7/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2669
    - accuracy: 0.9013
Epoch 8/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2556
    - accuracy: 0.9045
Epoch 9/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2462
    - accuracy: 0.9064
Epoch 10/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2372
    - accuracy: 0.9113
```

Με optimizer AMSGrad έχω:

```
Epoch 1/10
1875/1875 [==============================] - 9s 5ms/step - loss: 0.5023
    - accuracy: 0.8248
Epoch 2/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3761
    - accuracy: 0.8661
Epoch 3/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.3359
    - accuracy: 0.8776
Epoch 4/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3132
    - accuracy: 0.8853
Epoch 5/10
```

```
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2959
    - accuracy: 0.8909
Epoch 6/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2813
    - accuracy: 0.8970
Epoch 7/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2703
    - accuracy: 0.9008
Epoch 8/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2615
    - accuracy: 0.9030
Epoch 9/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2523
    - accuracy: 0.9064
Epoch 10/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2440
    - accuracy: 0.9087
```



Σχήμα 11: Loss & Accuracy

Υπολογίζω την ευκλείδια απόσταση μεταξύ των διαγραμμάτων και έχω:

```
    Euclidean distances between loss curves:
Adam vs SGD: 0.47037915557558463
Adam vs RMSprop: 0.07177903593668561
Adam vs Adagrad: 1.0700938463518048
Adam vs Adadelta: 3.464192311347625
Adam vs AdamW: 0.008707468059085828
Adam vs Nadam: 0.01299653800055993
Adam vs AMSGrad: 0.00827949273808489
SGD vs RMSprop: 0.41626618887437883
SGD vs Adagrad: 0.6020569491335517
SGD vs Adadelta: 3.006337845286188
SGD vs AdamW: 0.46510608467316356
SGD vs Nadam: 0.48129097228184115
```

```
SGD vs AMSGrad: 0.4637963777348849
RMSprop vs Adagrad: 1.0180847263366641
RMSprop vs Adadelta: 3.4151608190537677
RMSprop vs AdamW: 0.06614657965458229
RMSprop vs Nadam: 0.0779190605692856
RMSprop vs AMSGrad: 0.06428654602636544
Adagrad vs Adadelta: 2.4156772859800624
Adagrad vs AdamW: 1.0649750607408608
Adagrad vs Nadam: 1.0813261163959742
Adagrad vs AMSGrad: 1.0637606450701211
Adadelta vs AdamW: 3.4585389328443306
Adadelta vs Nadam: 3.474952149310836
Adadelta vs AMSGrad: 3.458350262769808
AdamW vs Nadam: 0.016753805537847215
AdamW vs AMSGrad: 0.008327664738688748
Nadam vs AMSGrad: 0.018876970095860163

Euclidean distances between accuracy curves:
Adam vs SGD: 0.13552188015224703
Adam vs RMSprop: 0.010034440691936961
Adam vs Adagrad: 0.29792873308897216
Adam vs Adadelta: 1.0794486277725308
Adam vs AdamW: 0.00250399680510968
Adam vs Nadam: 0.00632771680782254
Adam vs AMSGrad: 0.0027784887978899984
SGD vs RMSprop: 0.12709740359267774
SGD vs Adagrad: 0.16319016514483953
SGD vs Adadelta: 0.9494789413146559
SGD vs AdamW: 0.13636469484437674
SGD vs Nadam: 0.14095602151025685
SGD vs AMSGrad: 0.1363392093273245
RMSprop vs Adagrad: 0.28984692511737986
RMSprop vs Adadelta: 1.072386977727723
RMSprop vs AdamW: 0.011009995458672907
RMSprop vs Nadam: 0.014169333082400117
RMSprop vs AMSGrad: 0.010169070754006955
Adagrad vs Adadelta: 0.789141045694621
Adagrad vs AdamW: 0.29873064790878084
Adagrad vs Nadam: 0.3035162433874009
Adagrad vs AMSGrad: 0.29884552865987474
Adadelta vs AdamW: 1.0802391170477026
Adadelta vs Nadam: 1.0852788028889166
Adadelta vs AMSGrad: 1.080548564387552
AdamW vs Nadam: 0.005980802621722248
AdamW vs AMSGrad: 0.0039458839313897395
Nadam vs AMSGrad: 0.005669215113223351
```

Άρα έχω:

1. Similar Loss Curves:
    - Adam vs AdamW: 0.0087

- Adam vs AMSGrad: 0.0083
- AdamW vs AMSGrad: 0.0083

2. Similar Accuracy Curves:
   - Adam vs AdamW: 0.0025
   - Adam vs AMSGrad: 0.0028
   - AdamW vs AMSGrad: 0.0039

Η παρατηρούμενη ομοιότητα μεταξύ των τριών optimizers οφείλεται στις κοινές βασικές αρχές και στρατηγικές βελτιστοποίησης που χρησιμοποιούν. Οι optimizers αυτοί, προερχόμενοι από την οικογένεια του Adam, χρησιμοποιουν τεχνικές προσαρμοζόμενης ταχύτητας μάθησης που προσαρμόζουν δυναμικά τις ταχύτητες μάθησης για κάθε παράμετρο με βάση τα προηγούμενα gradients. Οι optimizers της οικογένειας Adam εξισορροπούν τα πλεονεκτήματα των AdaGrad και RMSProp, προσφέροντας robustness και αποδοτικότητα στη βελτιστοποίηση μοντέλων βαθιάς μάθησης. Ο AdamW, μια παραλλαγή του Adam, εισάγει weight decay απευθείας στα update rules, οπότε προσφέρει καλύτερα regularization και generalization. Επίσης, ο AMSGrad διευθετεί πιθανά προβλήματα του Adam εξασφαλίζοντας τα learning rates δεν εκτοξεύονται ποτέ, οδηγώντας σε πιο σταθερή συμπεριφορά σύγκλισης. Συνεπώς, η κοινή τους έμφαση σε adaptive learning rates και μηχανισμούς κανονικοποίησης προάγει συγκρίσιμη απόδοση όσον αφορά τη μείωση του σφάλματος και τη βελτίωση της ακρίβειας σε deep learning tasks.