Στοχαστική Ανάλυση Τυχαίου Σήματος με Θόρυβο

Ανδρέας Αυγούστης
Τμήμα Πληροφορικής
Ιόνιο Πανεπιστήμιο
Κέρκυρα, Ελλάδα
inf.bdn2202@ionio.gr

***Abstract*—Η επεξεργασία σήματος είναι ουσιαστικώς ένα διεπιστημονικό γνωστικό πεδίο, ορισμένο με αυστηρά μαθηματικά και με τις δικές του μεθοδολογίες και ορολογία. Οι εφαρμογές του είναι πάρα πολλές στις τεχνολογικές επιστήμες και βρίσκεται στη βάση τομέων όπως οι τηλεπικοινωνίες, ο αυτοματισμός, η επεξεργασία εικόνας, βίντεο και ήχου, η συμπίεση δεδομένων κλπ. Σε αυτή την εργασία θα επικεντρωθούμε στην ανάλυση,επεξεργασία και καθαρισμό ενός τυχαίου σήματος με χρήση στοχαστικών εξισώσεων και απλών μαθηματικών συναρτήσεων.**

***Index Terms*—στοχαστικές εξισώσης, μαθηματικές συναρτήσεις, σήματα, επεξεργασία σήματος**

I. Εισαγωγή

Επεξεργασία σήματος ορίζουμε την ανάλυση και τον χειρισμό σημάτων, όπου ως σήμα ορίζεται οποιαδήποτε συνάρτηση μεταξύ φυσικών ποσοτήτων. Η επεξεργασία σήματος είναι ουσιαστικώς ένα διεπιστημονικό γνωστικό πεδίο, ορισμένο με αυστηρά μαθηματικά και με τις δικές του μεθοδολογίες και ορολογία. Οι εφαρμογές του είναι πάρα πολλές στις τεχνολογικές επιστήμες και βρίσκεται στη βάση τομέων όπως οι τηλεπικοινωνίες, ο αυτοματισμός, η επεξεργασία εικόνας, βίντεο και ήχου, η συμπίεση δεδομένων κλπ. Χρησιμοποιείτε σε συστήματα τηλεπικοινωνιών, επεξεργασία σήματος λαμβάνει χώρα μόνο στο πρώτο επίπεδο του μοντέλου αναφοράς OSI, το φυσικό επίπεδο, και προαιρετικά στο έκτο και έβδομο επίπεδο του ίδιου μοντέλου. [1].

Στη θεωρία πιθανοτήτων και σε συναφή πεδία, μια στοχαστική ή τυχαία διαδικασία είναι ένα μαθηματικό αντικείμενο που συνήθως ορίζεται ως μια ακολουθία τυχαίων μεταβλητών, όπου ο δείκτης της ακολουθίας έχει την ερμηνεία του χρόνου. Οι στοχαστικές διαδικασίες χρησιμοποιούνται ευρέως ως μαθηματικά μοντέλα συστημάτων και φαινομένων που φαίνεται να ποικίλλουν με τυχαίο τρόπο. Παραδείγματα περιλαμβάνουν την ανάπτυξη ενός πληθυσμού βακτηρίων, ένα ηλεκτρικό ρεύμα που μεταβάλλεται λόγω θερμικού θορύβου ή την κίνηση ενός μορίου αερίου. Οι στοχαστικές διεργασίες έχουν εφαρμογές σε πολλούς κλάδους όπως η βιολογία, η χημεία, η οικολογία, η νευροεπιστήμη, η φυσική, η επεξεργασία εικόνας, η επεξεργασία σήματος , θεωρία ελέγχου, θεωρία πληροφοριών, επιστήμη υπολογιστών, κρυπτογραφία και τηλεπικοινωνίες. Επιπλέον, οι φαινομενικά τυχαίες αλλαγές στις χρηματοπιστωτικές αγορές έχουν παρακινήσει την εκτεταμένη χρήση στοχαστικών διαδικασιών στα χρηματοοικονομικά. [8].

Οι εφαρμογές και η μελέτη των φαινομένων έχουν με τη σειρά τους εμπνεύσει την πρόταση νέων στοχαστικών διεργασιών. Παραδείγματα τέτοιων στοχαστικών διεργασιών περιλαμβάνουν τη διαδικασία Wiener ή τη διαδικασία κίνησης Brown, που χρησιμοποιείται από τον Louis Bachelier για τη μελέτη των μεταβολών των τιμών στο. Αυτές οι δύο στοχαστικές διαδικασίες θεωρούνται οι πιο σημαντικές και κεντρικές στη θεωρία των στοχαστικών διεργασιών και ανακαλύφθηκαν επανειλημμένα και ανεξάρτητα. [3]

Σε αυτήν την εργασία, γίνεται παραγωγή τυχαίου σήματος θορύβου όπου με χρήση python [10] και απλών μαθηματικών συναρτήσεων [4] [5] γίνεται η επεξεργασία και ο καθαρισμός του.

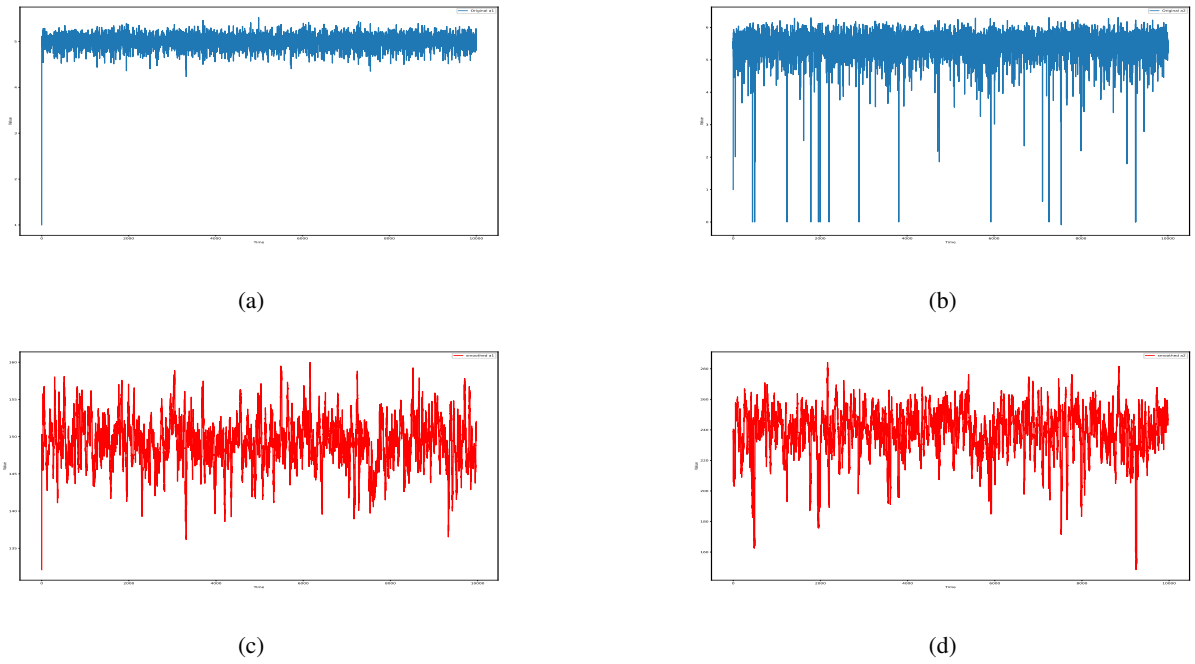
II. Μαθηματικές συναρτήσεις

Ο προσδιορισμός των νόμων που διέπουν τα διάφορα φαινόμενα και η κατανόηση της μορφής τους, δίνει συχνά τη δυνατότητα πρόβλεψης της μελλοντικής τους εξέλιξης.

Για φαινόμενα που περιγράφονται από ντετερμινιστικές διαδικασίες, αυτό μπορεί να γίνει με μεγαλύτερη ευκολία. Ωστόσο, τα περισσότερα φαινόμενα, εξελίσσονται στο χρόνο παρουσιάζοντας ένα βαθμό "τυχαιότητας" κι επομένως, για την περιγραφή τους θα πρέπει να βρεθεί κατάλληλο στοχαστικό μοντέλο. Μια στοχαστική διαδικασία που χρησιμοποιείται όλο και συχνότερα τα τελευταί- α χρόνια στη μοντελοποίηση- κυρίως οικονομικών δεδομένων, είναι η διαδικασία Ornstein-Uhlenbeck [5]. όπως ορίζεται :

$$dX_t = -\alpha X_t dt + \beta dW_t \tag{1}$$

, όπου a και β είναι σταθερές, Xt είναι ο χρόνος του θόρυβου , w είναι το βάρος του θόρυβου. Η παραπάνω στοχαστική διαδικασία μας επιτρέπει την παραγωγή και ανάλυση σημάτων. Κάθε σήμα έχει θόρυβο που δυσκολεύει την σωστή ανάλυση και κατανόηση του σήματος. Με βάση αυτής της υπόθεσης χρησιμοποιείται απλή τεχνική εξάλειψης θορύβου ή λεγόμενη εξομάλυνση του κινούμενου μέσου



(a) (b) (c) (d)

Fig. 1: (a) Αποτελέσματα της a μαζί με την smoothed για s=30. (b) Αποτελέσματα της a μαζί με την smoothed a για s=120. (c) Αποτελέσματα της smoothed a για s=30. (d) Αποτελέσματα της smoothed a για s=120.

όρου(Smoothing Moving Average) [4] που ορίζεται ως εξής :

$$\text{SMA} = \frac{X_1 + X_2 + X_3 + \ldots + X_n}{n} \tag{2}$$

όπου

- Το SMA αντιπροσωπεύει τον απλό κινούμενο μέσο όρο
- X1, X2, X3, ..., Xn είναι τα σημεία δεδομένων που θέλουμε να μετρήσουμε
- n είναι ο αριθμός των σημείων δεδομένων που υπολογίζετε κατά μέσο όρο (γνωστό και ως μέγεθος παραθύρου)

III. Υλοποίηση μεθοδολογίας

Στην περίπτωση της μη επιτυχής καθαρισμό θορύβου θα πρέπει να γίνει χρήση λογαριθμικών [6] και εκθετικών πράξεων [9] και συγκεκριμένα η εξή:

$$\log(a) = \log(b) + \log(c) \tag{3}$$

$$\text{a} = e^{\log(x)} \tag{4}$$

Εν συνεχεία των βασικών μαθηματικών συναρτήσεων θα γίνει υλοποίηση απλής εξίσωσης για την εξάληψη του θορύβου. Η εξίσωση που θα υλοποιήσουμε είναι η :

$$\alpha = \beta * \gamma \tag{5}$$

όπου

- το β θα είναι η εξίσωση της διαδικασίας Ornstein-Uhlenbeck [7]
- το γ θα είναι θόρυβος

Η εξίσωση που αναπαριστά το β είναι η Ornstein-Uhlenbeck. Το γ αναπαριστά τον περαιτέρω θόρυβο που πολλαπλασιάζουμε. Μέσω της εξίσωσης που είδαμε στο κεφάλαιο II μπορούμε να αναπαράξουμε τυχαίο σήμα. Για την αναπαρωγή τυχαίου σήματος χρησιμοποιούμε τυχαίους αριθμούς από το -1,1 όπως βλεπετε στον παρακάτω κομμάτι του κώδικα

```
t1 = np.zeros(10000)
t2 = np.zeros(10000)
b1 = np.zeros(10000)
b2 = np.zeros(10000)
a1 = np.zeros(10000)
a2 = np.zeros(10000)
a = 0.9
s1 = 30
s2 = 120
w1 = np.zeros(10000)
w2 = np.zeros(10000)
g = np.zeros(10000)
for i in range(10000):
    w1[i] = round(random.uniform(-1.00,1.00)↩
        ,1)
    w2[i] = round(random.uniform(-1.00,1.00)↩
        ,1)
    g[i] = round(random.uniform(-1.00,1.00)↩
        ,1)
window_size = 30
```

Στο πρώτο στάδιο δημιουργούμε πίνακες τιμών με 10000 θέσης και αρχικοποιούμε τα βάρη, τον περαιτέρω θόρυβο και το παράθυρο για να χρησιμοποιήσουμε για την εξομάλυνση του κινούμενου μέσου όρου(Smoothing Moving Average). Εν συνεχεία εκτελούμε την εξίσωση για κάθε τιμή του i

και πολλαπλασιάζουμε τον θόρυβο(χρησιμοποιείται και για τις 2 σταθερες s1,s2). Το αποτέλεσμα αυτού το κάνουμε εξομάλυνση του κινούμενου μέσου όρου όπως βλέπετε στον παρακάτω κώδικα και στην εικόνα 1 .

```
t1[0] = 1
for i in range(1,10000):
    if i != 0 :
        t1[i] = (-a * t1[i-1] + s1 * w1[i])
        b1[i] = t1[i]
        a1[i] = b1[i]*g[i]
smoothed_a1 = np.convolve(a1, np.ones(↩
    window_size) / window_size, mode='valid')

t2[0] = 1
for i in range(1,10000):
    if i != 0 :
        t2[i] = (-a * t2[i-1] + s2 * w2[i])
        b2[i] = t2[i]
        a2[i] = b2[i]*g[i]
smoothed_a2 = np.convolve(a2, np.ones(↩
    window_size) / window_size, mode='valid')
```

Αφού δημιουργηθούν οι εξισώσεις, τα αποτελέσματα τους τα χρησιμοποιούμε για την δημιουργία τις διαφοράς μεταξύ της ήδη υπάρχουσας κατάσταση θορύβου με την προηγούμενη όπως φαίνεται στον παρακάτω κώδικα. Το αποτέλεσμα του θα πρέπει να το εξομαλύνουμε με χρήση του κινούμενου μέσου όρου και στην εικόνα 2.Για την επιτυχή εξομάλινση του θορύβου στην περίπτωση τη δικιά μας θα πρέπει να χρησιμοποιήσουμε την ιδιότητα του λογαρίθμου [2] :

$$\log(\alpha) = \log(\beta) + \log(\gamma) \tag{6}$$

Το αποτέλεσμα τις παραπάνω εξίσωσης θα πρέπει να το φέρουμε στην αρχική του μορφή ώστε να μπορέσουμε να το κάνουμε correlated. Σε αυτήν την περίπτωση θα χρησιμοποιήσουμε την ιδιότητα της εκθετικής συνάρτησης με εκθέτη λογάριθμο. Την ιδιότητα αυτή θα την χρησιμοποιήσουμε εφόσον χρησιμοποιήσουμε στην smoothed του α :

$$\alpha = e^{\log(\alpha)} \tag{7}$$

```
diff1 = [a1[i] - a1[i-1] for i in range(1,↩
    len(a1))]
diff2 = [a2[i] - a2[i-1] for i in range(1,↩
    len(a2))]
smoothed_diff1 = np.convolve(diff1, np.ones(↩
    window_size) / window_size, mode='valid')
smoothed_diff2 = np.convolve(diff2, np.ones(↩
    window_size) / window_size, mode='valid')
```

Τέλος δημιουργούμε το διάγραμμα autocorrelation για s = 30 και s = 120.Το aucorrelation μας δείχνει το πόσο γρήγορα πάει στο 0 ο θόρυβος. Τοποθετούμε το autocorrelation με lag = 50 όπως βλέπουμε στην εικόνα 3 και παρατηρούμε την ταχύτητα μηδενισμού του θορύβου.

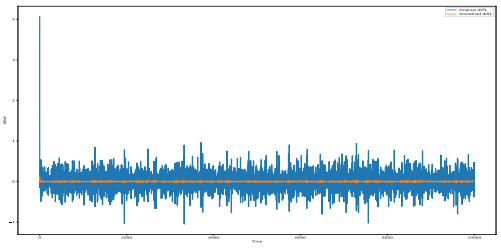
IV. Συμπέρασμα

Με βάση την παραπάνω υλοποίηση και με χρήση βασικών στοχαστικών διαδικασιών παρατηρούμε ότι το σήμα ακόμα κ με εξομαλυνση με κινούμενο μέσω όρο δεν καθαρίζει τελειώς το σήμα εκτός από την χρήση της διαφοράς όπου φαίνεται ο καθαρισμος του σήματος. Με αυτές τις μεθόδους μπορούμε να επιτύχουμε ικανοποιητικά αποτελέσματα στην ανάλυση και επεξεργασία σήματος με σκοπό την εξομάλυνση του θορύβου.

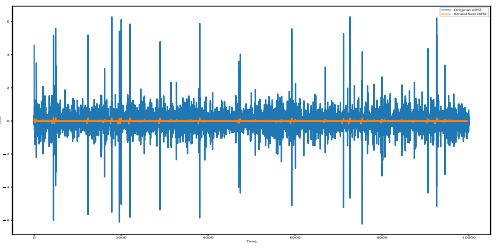
V. Συνολικός Κώδικας Υλοποίησης

Παρακάτω απεικονίζεται ο συνολικός κώδικας υλοποίησης της μεθοδολογίας.

```
import numpy as np
import matplotlib.pyplot as plt
import random
import csv
from statsmodels.graphics.tsaplots import ↩
    plot_acf
plt.rcParams["figure.figsize"] = (20,15)
t1 = np.zeros(10000)
t2 = np.zeros(10000)
b1 = np.zeros(10000)
b2 = np.zeros(10000)
a1 = np.zeros(10000)
a2 = np.zeros(10000)
a = 0.8
s1 = 30
s2 = 120
for i in range(10000):
    t1[i] = 1
    t1[i] = 1
```

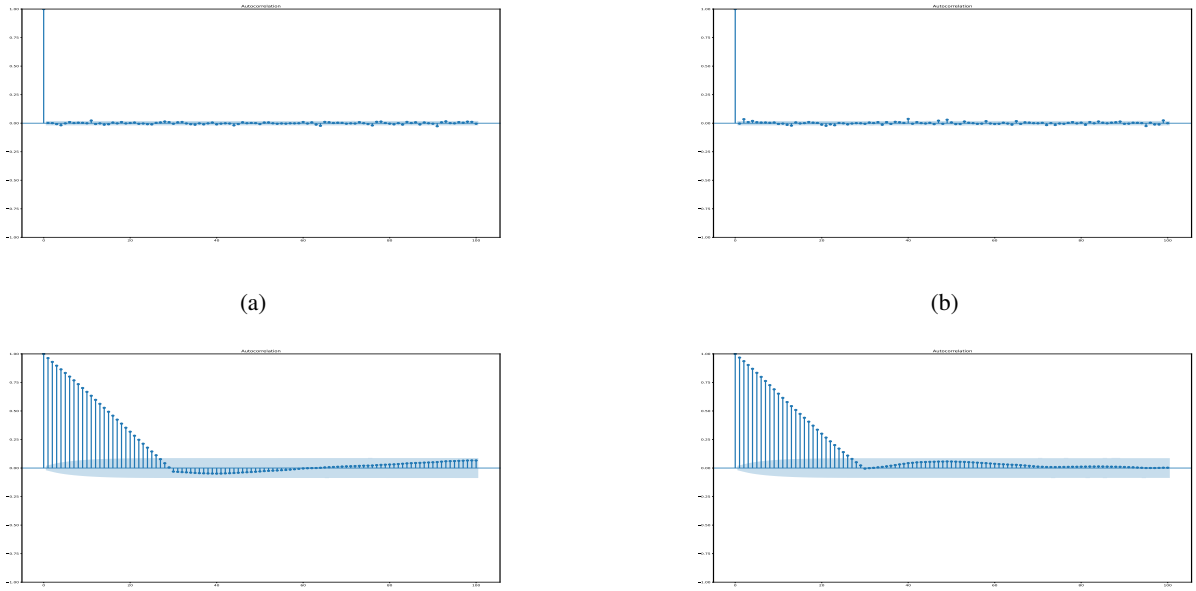


(a)



(b)

Fig. 2: (a) Αποτελέσματα της διαφοράς a μαζί με την smoothed για s=30. (b) Αποτελέσματα της διαφοράς a μαζί με την smoothed a για s=120.



(a) (b)

(c) (d)

Fig. 3: (a) Autocorrelation για τα αποτελέσματα της a με s=30. (b) Autocorrelation για τα αποτελέσματα της a με s=120. (c) Autocorrelation για τα αποτελέσματα της smoothed a με s= 30. (d) Autocorrelation για τα αποτελέσματα της smoothed a με s=120.

```
    b1[i] = 1
    b2[i] = 1
    a1[i] = 1
    a2[i] = 1
w1 = np.zeros(10000)
w2 = np.zeros(10000)
g1 = np.zeros(10000)
g2 = np.zeros(10000)
for i in range(10000):
    w1[i] = round(random.uniform(-1.00,1.00)↩
        ,1)
    w2[i] = round(random.uniform(-1.00,1.00)↩
        ,1)
    g1[i] = round(random.uniform(-1.00,1.00)↩
        ,1)
    g2[i] = round(random.uniform(-1.00,1.00)↩
        ,1)
window_size = 30
for i in range(1,10000):
    if i != 0 :
        t1[i] = (-a * t1[i-1] + s1 * w1[i])
        b1[i] = t1[i]
        a1[i] = (b1[i] * g1[i]) +150
        a1[i] = np.log(a1[i])

nan_indices = np.isnan(a1)
a1[nan_indices] = 0
smoothed_a1 = np.convolve(a1, np.ones(↩
    window_size) / window_size, mode='valid')
smoothed_a1 = np.exp(smoothed_a1)
plt.plot(a1, label='Original a1')
plt.xlabel('Time')
plt.ylabel('Value')
plt.legend()
plt.savefig('a1.pdf', format='pdf')
plt.show()
plt.plot(smoothed_a1, label='smoothed a1',↩
    color = 'red')
plt.xlabel('Time')
plt.ylabel('Value')
plt.legend()
plt.savefig('smoothed_a1.pdf', format='pdf')
plt.show()
diff1 = [a1[i] - a1[i-1] for i in range(1,↩
    len(a1))]
for i in range(1,10000):
    if i != 0 :
        t2[i] = -a*t2[i-1] + s2*w2[i]
        b2[i] = t2[i]
        a2[i] = (b2[i] * g2[i] ) + 250
        a2[i] = np.log(a2[i])

nan_indices = np.isnan(a2)
a2[nan_indices] = 0
smoothed_a2 = np.convolve(a2, np.ones(↩
    window_size) / window_size, mode='valid')
smoothed_a2 = np.exp(smoothed_a2)
plt.plot(a2, label='Original a2')
plt.xlabel('Time')
plt.ylabel('Value')
plt.legend()
plt.savefig('a2.pdf', format='pdf')
plt.show()
plt.plot(smoothed_a2, label='smoothed a2',↩
    color = 'red')
plt.xlabel('Time')
plt.ylabel('Value')
plt.legend()
```

```
plt.savefig('smoothed_a2.pdf', format='pdf')
plt.show()
diff2 = [a2[i] - a2[i-1] for i in range(1,↩
    len(a2))]
smoothed_diff1 = np.convolve(diff1, np.ones(↩
    window_size) / window_size, mode='valid')
smoothed_diff2 = np.convolve(diff2, np.ones(↩
    window_size) / window_size, mode='valid')
plt.plot(diff1, label='Original diff1')
plt.plot(smoothed_diff1, label='Smoothed ↩
    diff1')
plt.xlabel('Time')
plt.ylabel('Value')
plt.legend()
plt.savefig('diff1.pdf', format='pdf')
plt.show()
plt.plot(diff2, label='Original diff2')
plt.plot(smoothed_diff2, label='Smoothed ↩
    diff2')
plt.xlabel('Time')
plt.ylabel('Value')
plt.legend()
plt.savefig('diff2.pdf', format='pdf')
plt.show()
plot_acf(a1,lags=100)
plt.savefig('autoc1.pdf', format='pdf')
plt.show()
plot_acf(smoothed_a1,lags=100)
plt.savefig('auto1.pdf', format='pdf')
plt.show()
plot_acf(a2,lags=100)
plt.savefig('autoc2.pdf', format='pdf')
plt.show()
plot_acf(smoothed_a2,lags=100)
plt.savefig('auto2.pdf', format='pdf')
plt.show()
```

REFERENCES

[1] Σοφία Εφραιμία Βρεττού. *στοχαστική προσομοίωση της φόρτισης ανέμου και κυμάτων στα υπεράκτια αιολικά πάρκα*. 2023.

[2] Κωνσταντίνος Δραζιώτης. *παραγοντοποίηση & διακριτός λογάριθμος*. 2022.

[3] Παπαϊωάννου Γεώργιος. *Στοχαστικά Μοντέλα Συνεχούς Χρόνου για την Αποτίμηση Αξιογράφων που Ενέχουν Πιστωτικό Κίνδυνο*. PhD thesis, University of Piraeus (Greece), 2022.

[4] Rob J Hyndman. Moving averages., 2011.

[5] Ross A Maller, Gernot Müller, and Alex Szimayer. Ornstein–uhlenbeck processes and extensions. *Handbook of financial time series*, pages 421–437, 2009.

[6] Μανώλης Δ Μαραγκουδάκης. *διακριτός λογάριθμος*. B.S. thesis, 2014.

[7] Μαρία Νικολάου Μιχάλη. *Διαδικασία Ornstein-Uhlenbeck*. PhD thesis, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, 2019.

[8] Κυριάκος Μουστάκας. *Διαγνωστική μελέτη για την ακρίβεια εκτίμησης παραμέτρων σε γραμμικά και μη-γραμμικά μοντέλα γεωδαιτικών χρονοσειρών θέσης*. PhD thesis, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, 2022.

[9] Βασιλική Πλεύρη. *Η συμπεριφορά του επενδυτή με εκθετική συνάρτηση χρησιμότητας*. 2020.

[10] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.