Εφαρμογή του αλγόριθμου Stochastic Outbreak σε σεισμολογικά δεδομένα

Νικόλαος Αυγούστης

30 Ιανουαρίου 2020

Περιεχόμενα

1 Το πρόβλημα

Το πρόβλημα με το οποίο ασχολείται η εργασία είναι η πρόβλεψη σεισμών σε πολύ αρχικά στάδια με τη χρήση στατιστικών μεθόδων βασισμένη σε μετρήσεις σεισμογράφων. Στο πλαίσιο αυτό υλοποιήθηκε ο αλγόριθμος Stochastic Outbreak που προτάθηκε από τον αναπληρωτή καθηγητή Μάρκο Αυλωνίτη σε παλαιότερη δημοσίευση του [1].

2 Σκιαγράφηση Αλγορίθμου

Η βασική ιδέα του αλγορίθμου είναι επιλογή ενός παραθύρου της χρονοσειράς, αφού την έχουμε μετατρέψει σε στατική παίρνοντας τις διαφορές των στοιχείων της, δεδομένων αρκετά μεγάλου μήκους ώστε οι μέθοδοί της στατιστικής να λειτουργούν και μετακινώντας το κάθε φορά πάνω από τα δεδομένα να ελέγχουμε την ύπαρξη ή μη θορύβου στο παράθυρο. Αυτό επιτυγχάνεται με τη χρήση της τεχνικής κινητού μέσου σε παράθυρο το οποίο καθορίζεται από τα μέγεθος του δείγματος (για παράδειγμα σε δείγμα 100 τουλάχιστον δεδομένων χρησιμοποιούμε τουλάχιστον 20 κινητούς μέσους) και δημιουργώντας από αυτούς τη συνάρτηση διακύμανσης η οποία ορίζεται ως η διακύμανση του κάθε εξομαλυμένου τμήματος προς τη διακύμανση του αρχικού. Αν η τελευταία ακολουθεί κατανομή Power law τότε τα δεδομένα στα συγκεκριμένο διάστημα είναι θόρυβος και προχωράμε στο επόμενο. Αν όμως αποκλίνει από Power law κατανομή το προχωράμε στον επόμενο

έλεγχο ο οποίος προτείνεται στη δημοσίευση [1] και ουσιαστικά μας λέει να λογαριθμισουμε τα δεδομένα μας και να αφαιρέσουμε από αυτά κατάλληλη παράμετρο ανάπτυξης η οποία προκύπτει κάθε φορά από τα δεδομένα. Αν τώρα η συνάρτηση διακύμανσης αυτών ακολουθεί κατανομή Power law έχουμε ξεφύγει από τον θόρυβο και στα δεδομένα μας υπάρχει κάποιο σήμα. Στην επόμενη ενότητα παρατίθεται ο κώδικας σε γλωσσά python που χρησιμοποιείται.

3 Κώδικας σε Python

```
def VarianceFunction(data):
    data = pd.Series(data)
    T = [i for i in range(2,22,2)]
    var0 = np.var(data)
    result = [np.var((data.rolling(window=t).mean())
                     .dropna())/var0 for t in T]
    return [result,T]
```

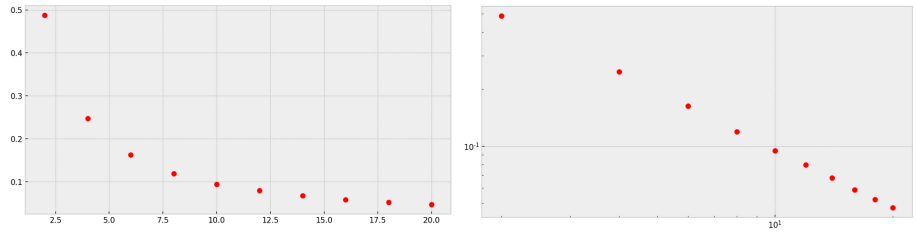
Figure 1: Κώδικας παραγωγής της συνάρτησης διακύμανσης.

```
def SO(data,window,step):
    rc = []
    L = pd.Series(data)
    start = 0
    end = start + window
    while(end <= len(L)):
        test = L[start:end]
        varfunc1 = VarianceFunction(test)
        CorCoef1 = np.corrcoef(np.log(varfunc1[0]), np.log(varfunc1[1]))[0,1]
        if(np.abs(CorCoef1) < 0.95):
            minval = np.min(test1)
            l = (np.max(test1) - np.min(test))/np.min(test)
            transf_data = [np.log(t+1-minval) - l for t in test]
            varfunc3 = VarianceFunction(transf_data)
            CorCoef2 = np.corrcoef(np.log(varfunc3[0])
                                   , np.log(varfunc3[1]))[0,1]
            if( np.abs(CorCoef2) > 0.99):
                rc.append(test)
                break
        start += step
        end += step
    return rc
```

Figure 2: Κώδικας υλοποίησης αλγορίθμου.

Παραπάνω φαίνονται ο κώδικας υλοποίησης του αλγορίθμου. Αυτό που πρέπει να τονιστεί είναι ο τρόπος που γίνεται ο έλεγχος για το αν η συνάρτηση διακύμανσης

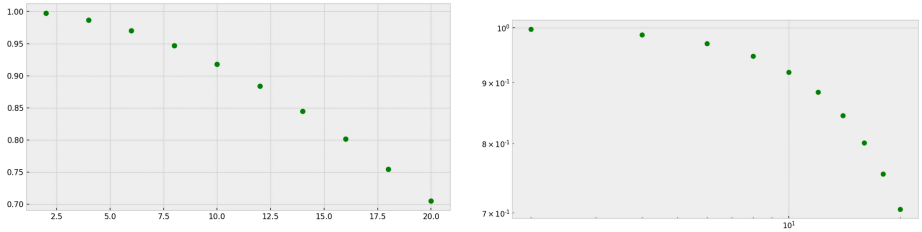
αποκλίνει ή όχι από την κατανομή Power law . Για αυτό τον έλεγχο τα δεδομένα και ο άξονας μετασχηματίζονται σε λογαριθμική κλίμακα και εκεί ελέγχεται ο συντελεστής συσχέτισης του Pearson (r), ο οποίος δηλώνει πόσο δυνατή είναι η γραμμική εξάρτηση των δυο μεταβλητών. Επειδή μας ενδιαφέρει η παραμικρή απόκλιση από αυτό θεωρούμε ότι αποκλίνουμε από κατανομή Power law αν ο συντελεστής πέσει κάτω του 0,95 κατά απόλυτη τιμή.



(α′) Συνάρτηση διακύμανσης (r = -0.81)

(β′) Συνάρτηση διακύμανσης σε λογαριθμημένους άξονες (r = -0.999)

Σχήμα 3: Τυχαία δεδομένα



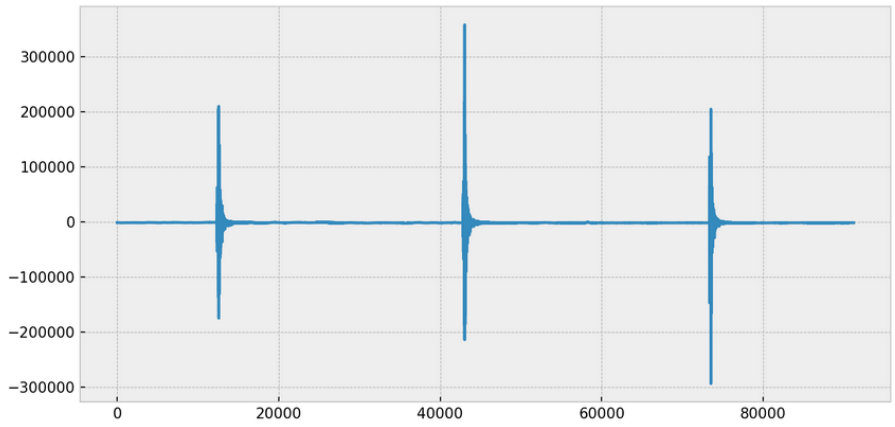
(α′) Συνάρτηση διακύμανσης (r = -0.982)

(β′) Συνάρτηση διακύμανσης σε λογαριθμημένους άξονες (r = -0.861)

Σχήμα 4: Δεδομένα με σήμα

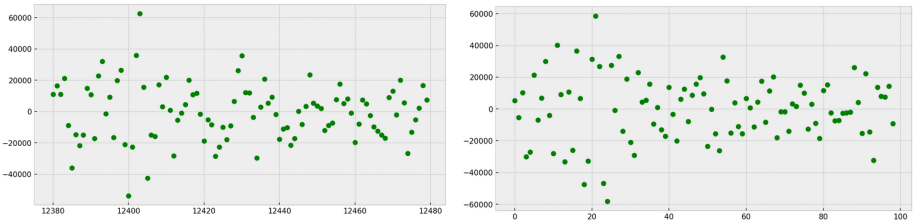
4 Τα δεδομένα

Παρακάτω δίνονται τα δεδομένα στα οποία εφαρμόστηκε ο αλγόριθμος.



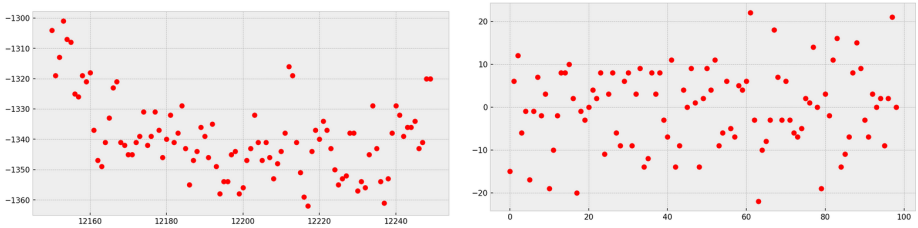
Σχήμα 5: Αρχική εικόνα των δεδομένων.

Στο σχήμα 5 φαίνεται η αρχική όψη των δεδομένων η οποία αρχικά μοιάζει να είναι αρκετά καλή για την εφαρμογή του αλγορίθμου.



(α΄) Περιοχή στα δεδομένα με σήμα. (β΄) Διαφορές σήματος.

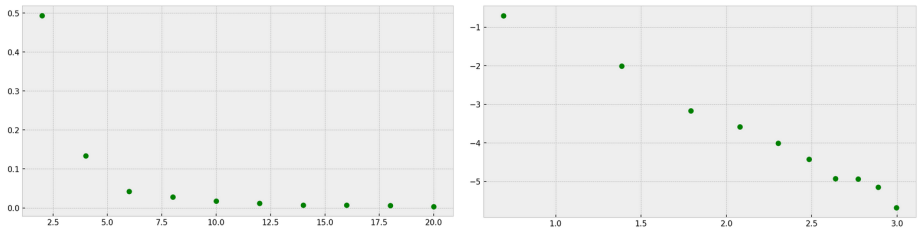
Σχήμα 6: Δεδομένα με σήμα.



(α΄) Περιοχή στα δεδομένα με θόρυβο. (β΄) Διαφορές θορύβου.

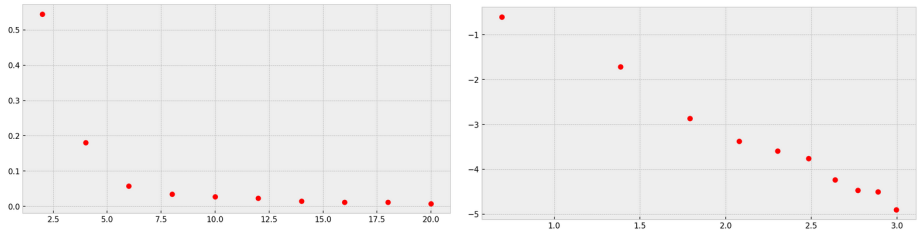
Σχήμα 7: Δεδομένα με χωρίς σήμα.

Από τα σχήματα 6 και 7 είναι προφανές ότι δεν φαίνεται διαφορά, τουλάχιστον οπτικά, της περιοχής του σήματος και του θορύβου. Οπότε ας δούμε οπτικά τις συναρτήσεις διακύμανσης των παραπάνω.



(α′) Συνάρτηση διακύμανσης (r = -0.669). (β′) Λογαριθμημένη συνάρτηση (r = -0.996).

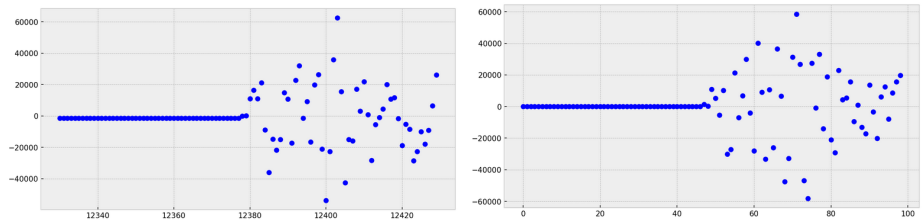
Σχήμα 8: Δεδομένα με σήμα.



(α′) Συνάρτηση διακύμανσης (r = -0.691). (β′) Λογαριθμημένη συνάρτηση (r = -0.994).

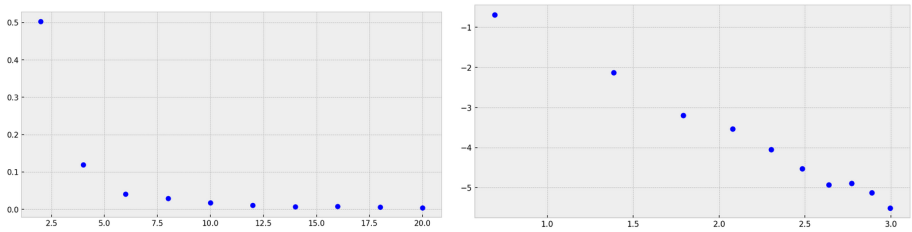
Σχήμα 9: Δεδομένα θορύβου.

Έχουμε τελικά και επιβεβαίωση από τη θεωρία ότι και τα δυο τμήματα από τα δεδομένα φαίνονται ως θόρυβος απλά το ένα έχει μεγαλύτερο πλάτος, οπότε η εφαρμογή του αλγορίθμου δεν περιμένουμε να δώσει ικανοποιητικά αποτελέσματα. Αλλά ας δούμε και την περιοχή δεδομένων στην οποία γίνεται η αλλαγή μεταξύ θορύβου και σήματος.



(α′) Δεδομένα. (β′) Διαφορές.

Σχήμα 10: Δεδομένα μετάβασης.



(α′) Συνάρτηση Διακύμανσης (r = -0.655).

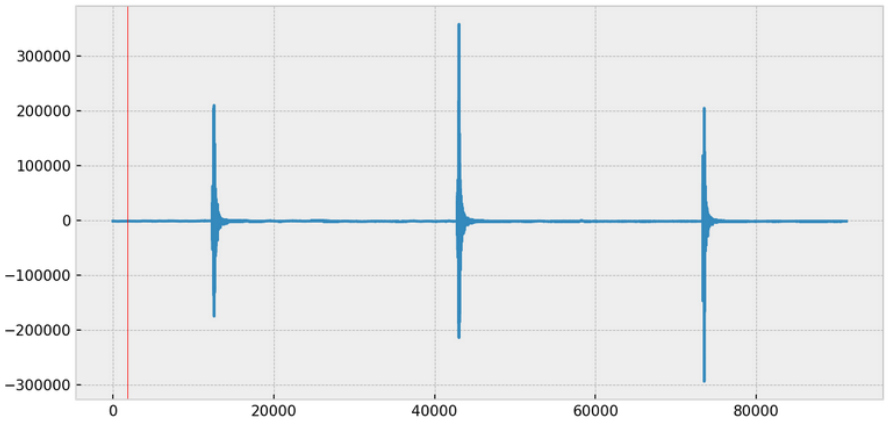
(β′) Λογαριθμημένη συνάρτηση (r = -0.996).

Σχήμα 11: Συνάρτηση διακύμανσης μετάβασης.

Και πάλι από τη θεωρία επιβεβαιωνόμαστε ότι ακόμα και εκεί που γίνεται η μετάβαση από το θόρυβο στο σήμα τα δεδομένα φαίνονται ως θόρυβος. Οπότε στα συγκεκριμένα δεδομένα τα αποτελέσματα δεν περιμένουμε να είναι ικανοποιητικά.

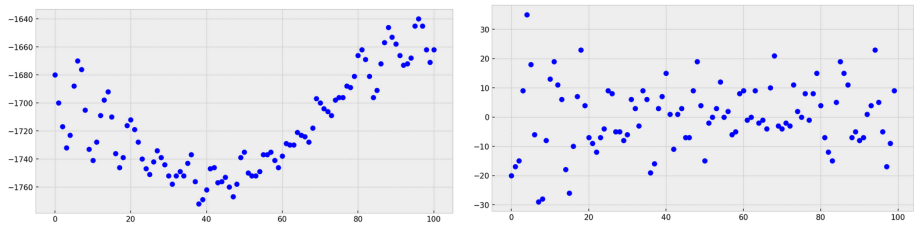
5 Αποτελέσματα στα δεδομένα

Η εφαρμογή του αλγορίθμου στα παραπάνω δεδομένα όπως περιμέναμε δεν έδωσε ικανοποιητικά αποτελέσματα λόγο της φύσης των δεδομένων, μας προειδοποιεί για έξαρση σε ένα σημείο στο οποίο είναι ξεκάθαρο ότι είναι θόρυβος όπως φαίνεται στο παρακάτω σχήμα 12.



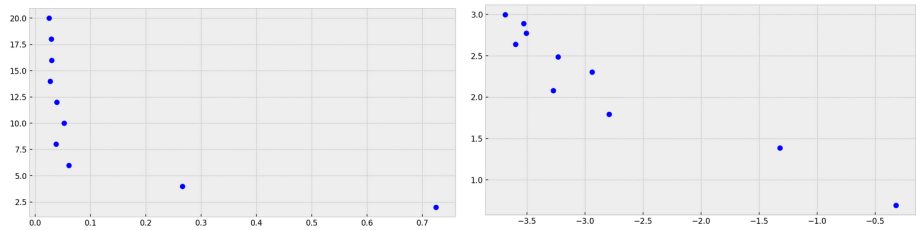
Σχήμα 12: Αποτελέσματα εφαρμογής στα δεδομένα.

Παρόλο που το αποτέλεσμα είναι προφανώς λάθος ας δούμε την περιοχή που ο αλγόριθμος επέλεξε.



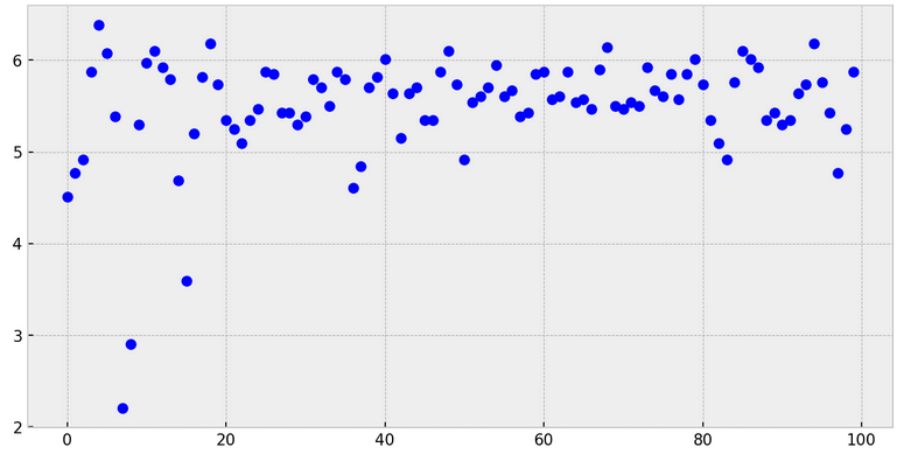
(α′) Περιοχη δεδομενων. (β′) Διαφορες.

Τα δεδομένα σε αυτή την περιοχή είναι θόρυβος όπως φαίνεται και από την εικόνα τους και από τη συνάρτηση διακύμανσης η οποία στον λογαριθμικό μετασχηματισμό της γίνεται ευθεία (Σχημα 14) αλλά όχι όπως θα περιμέναμε. Παρόλα αυτά λόγω της χρήσης του συντελεστή Pearson ο οποίος απλά ελέγχει το πόσο γραμμικά είναι τα δεδομένα και δεν μας λέει τίποτα για το σχήμα τους περνάει τον πρώτο έλεγχο και προφανώς και τον δεύτερο διότι μετασχηματίζοντας το θόρυβο παίρνω πάλι θόρυβο (Σχήμα 15). Οπότε παίρνω και την λάθος απάντηση από τον αλγόριθμο.

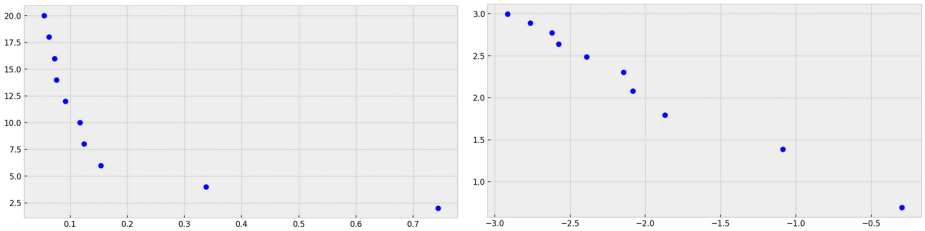


(α′) Συνάρτηση διακύμανσης (r = -0.675). (β′) Λογαριθμημένη συνάρτηση (r = -0.948).

Σχήμα 14



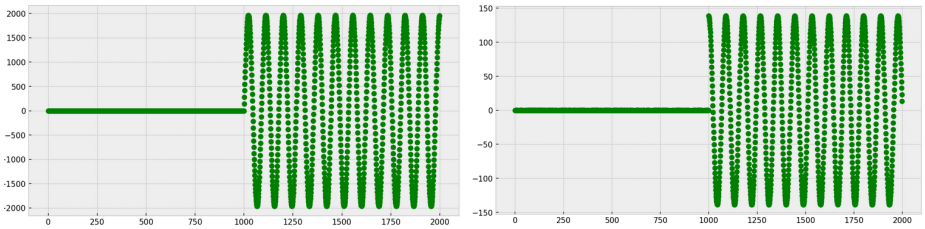
Σχήμα 15: Μετασχηματισμένα δεδομένα.



(α′) Συνάρτηση διακύμανσης (r = -0.748). (β′) Λογαριθμημένη συνάρτηση (r = -0.990).

6 Αποτελέσματα Σε άλλα δεδομένα

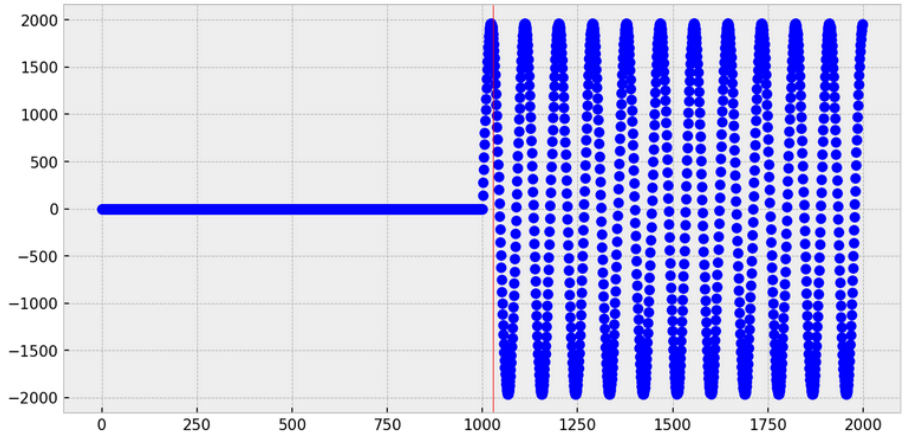
Επειδή τα δεδομένα που έτρεξε αρχικά ο αλγόριθμος δεν έδωσαν ικανοποιητικά αποτελέσματα κατασκευάστηκε ένα τεχνητό σετ δεδομένων για να ξαναδοκιμαστεί ο αλγόριθμος.



(α′) Τεχνητά δεδομένα. (β′) Διαφορές.

Σχήμα 17: Συνάρτηση διακύμανσης μετάβασης.

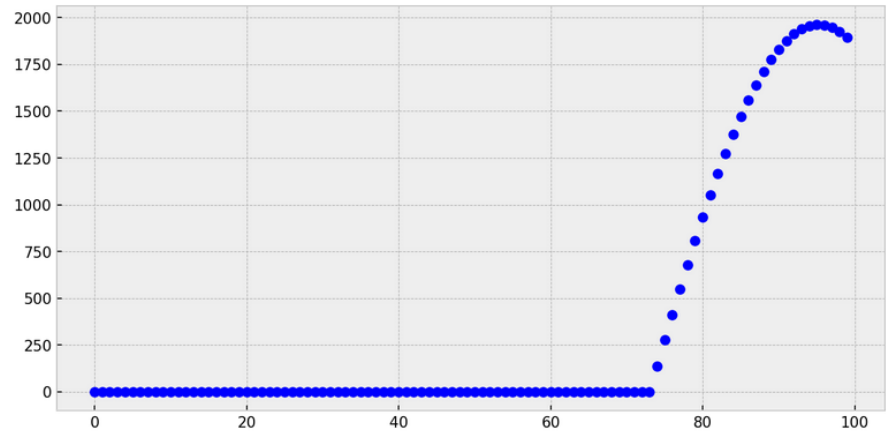
Σε αυτά τα δεδομένα ο αλγόριθμος εντόπισε έξαρση στο φαινόμενο στη θέση 1027 όπως φαίνεται στο παρακάτω σχήμα 18



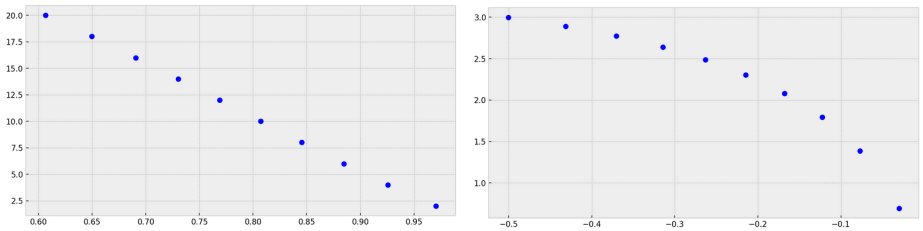
Σχήμα 18: Αποτέλεσμα Του αλγορίθμου.

Ο αλγόριθμος έτρεξε με παράθυρο 100 δεδομένων οπότε αξίζει να δούμε το

συγκεκριμένο παράθυρο δεδομένων



Σχήμα 19: Περιοχή έξαρσης.

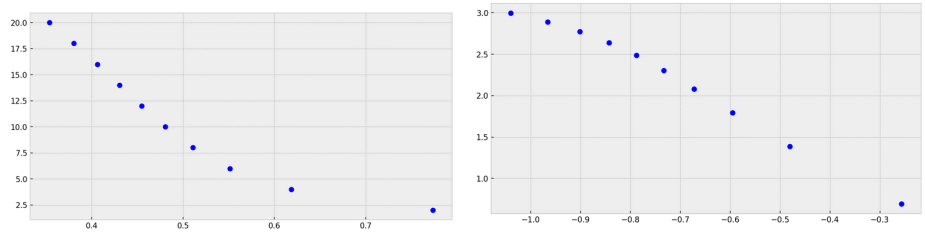


(α΄) Συνάρτηση Διακύμανσης (r = -0.999).

(β΄) Λογαριθμημένη Συνάρτηση (r = -0.9333).

Σχήμα 20

Όπως φαίνεται στο σχήμα 20 έχουμε αρκετή απόκλιση από Power law οπότε μπορούμε να προχωρήσουμε σε περαιτέρω έλεγχο.



(α΄) Συνάρτηση Διακύμανσης (r = -0.936).

(β΄) Λογαριθμημένη Συνάρτηση (r = -0.993).

Σχήμα 21

Μετασχηματίζοντας τα δεδομένα και ελέγχοντας τώρα τη συνάρτηση διακύμαν-

σης έχω τα αποτελέσματά που φαίνονται στο σχήμα 21. Άρα μπορώ να επιβεβαιώσω πλέον ότι βρίσκομαι σε σημείο έξαρσης του φαινομένου.

References

[1] Avlonitis, M., On the problem of early detection of users interaction outbreaks via stochastic differential models, Eng. Appl. Artif. Intel. (2016)