



ΠΑΝΕΠΙΣΤΗΜΙΟ
ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

School of Engineering
Computer Engineering & Informatics Department

Post-graduate Program “Computer Science and Engineering”

Master Thesis

*Scisola: Automatic Moment Tensor
solution for SeisComP3*

Nikolaos Triantafyllis

ID 926

Supervisor

Professor Pavlidis Georgios

Co-supervisors

Assistant Professor Efthimios Sokos, Geology Department

PhD Candidate Aristidis Ilias

Patras, 2014

A summary in Greek follows.

Ακολουθεί περίληψη στα Ελληνικά.

Πανεπιστήμιο Πατρών, Πολυτεχνική Σχολή,

Τμήμα Μηχανικών Η/Υ & Πληροφορικής

Μεταπτυχιακό Πρόγραμμα «Επιστήμη και Τεχνολογία Υπολογιστών»

Μεταπτυχιακή Διπλωματική Εργασία

***Scisola: Υπολογισμός του Τανυστή Σεισμικής Ροπής
για το λογισμικό SeisComP3***

Νικόλαος Τριανταφύλλης AM 926

Επιβλέπων: Καθηγητής Παυλίδης Γεώργιος

Συνεπιβλέποντες: Επίκουρος Καθηγητής Ευθύμιος Σώκος, Τμήμα Γεωλογίας

Υποψήφιος Διδάκτορας Αριστείδης Ηλίας

Πάτρα, 2014

Τανυστής Σεισμικής Ροπής

Ο Τανυστής Σεισμικής Ροπής (ΤΣΡ) είναι μια μαθηματική αναπαράσταση της ολίσθησης κατά τη διάρκεια του σεισμού και σχετίζεται άμεσα με τη γεωμετρία του ρήγματος και το μέγεθος του σεισμού. Οι ΤΣΡ χρησιμοποιούνται σε ένα ευρύ φάσμα ερευνητικών θεμάτων της σεισμολογίας όπως τη σεισμοτεκτονική, τη συσχέτιση μεταξύ σεισμών με βάση την κλίμακα υπολογισμών των μεγεθών τους και την αντιστροφή του τανυστή της τάσης (stress inversion), την μοντελοποίηση θαλασσίων κυμάτων βαρύτητας (tsunami), των προσδιορισμό των καταστροφών κτλ. και ως εκ τούτου είναι σημαντικός ο γρήγορος και αξιόπιστος υπολογισμός τους.

Διεθνής εμπειρία

Ο υπολογισμός του ΤΣΡ μπορεί να χωριστεί σε δύο κατηγορίες, ανάλογα με την προέλευση των δεδομένων:

1. δεδομένα από παγκόσμια δίκτυα
2. δεδομένα από τοπικά ή περιφερειακά δίκτυα

Έχουν υλοποιηθεί διάφορες εργασίες που πραγματοποιούν την αυτοματοποίηση της διαδικασίας της αντιστροφής κυρίως όσον αφορά στην πρώτη κατηγορία, όπως είναι το Global CMT (<http://www.globalcmt.org/>), καθώς και άλλα. Ωστόσο, όσον αφορά στη δεύτερη κατηγορία, οι προσπάθειες είναι αρκετά λιγότερες και στοχευμένες στο εκάστοτε δίκτυο, χωρίς ουσιαστική δυνατότητα να εφαρμοστούν γενικά. Ωστόσο, πρόσφατα δημιουργήθηκε ένα λογισμικό που υπολογίζει αυτόματα τον Τανυστή -με την ονομασία amt- (Triantafyllis et al., 2013), συνδέοντας διάφορα εργαλεία λογισμικού μεταξύ τους όπως το ISOLA που μπορεί να υπολογίσει τον Τανυστή, καθώς και το nmxp tool με το οποίο παρέχεται η δυνατότητα επικοινωνίας με τους NAQ servers -συστήματα που αποθηκεύουν σεισμική πληροφορία-. Αυτή η αρχική υλοποίηση άνοιξε το δρόμο για μια αυτόματη διαδικασία υπολογισμού του ΤΣΡ η οποία να είναι στενά συνδεδεμένη με

ένα ευρέως γνωστό λογισμικό αυτόματης επεξεργασίας σεισμικών δεδομένων, το SeisComP3.

Στόχοι της μεταπτυχιακής διπλωματικής εργασίας

Η διπλωματική αυτή εργασία έχει ως στόχο να δημιουργήσει ένα λογισμικό (Scisola), το οποίο θα παρέχει τις ακόλουθες λειτουργίες:

1. αυτοματοποίηση του υπολογισμού του Τανυστή Σεισμικής Ροπής με διασύνδεση στο ευρέως γνωστό λογισμικό, SeisComP3
2. αποθήκευση των αποτελεσμάτων σε βάση δεδομένων
3. δυνατότητα παραμετροποίησης του λογισμικού από το χρήστη με χρήση γραφικού περιβάλλοντος (configuration)
4. δυνατότητα προβολής των αποτελεσμάτων με χρήση γραφικού περιβάλλοντος (overview)
5. δυνατότητα τροποποίησης των αποτελεσμάτων βάση επιλογής του χρήστη με χρήση γραφικού περιβάλλοντος (revision)

Εργαλεία συστήματος

Η υλοποίηση του SCISOLA περιλαμβάνει συγγραφή κώδικα σε Python μαζί με χρήση βιβλιοθηκών ανοιχτού κώδικα όπως τις ObsPy, matplotlib, PyQt, MySQLdb, psycopg2 κ.ά., αλλά και διάφορων άλλων εργαλείων.

Συγκεκριμένα τα βασικά εργαλεία και βιβλιοθήκες που χρησιμοποιήθηκαν είναι τα ακόλουθα:

- SeisComP3
- ISOLA
- MySQL
- Python programming language
- Python packages:
 - ObsPy
 - Matplotlib
 - Numpy
 - Subprocess

- Multiprocessing
- PyQt4
- MySQLdb
- psycopg2

Στη συνέχεια ακολουθεί μια συνοπτική περιγραφή των παραπάνω εργαλείων και βιβλιοθηκών.

Το SeisComP3 είναι ένα λογισμικό που αναπτύχθηκε πάνω στο λειτουργικό σύστημα Linux τα τελευταία δέκα περίπου χρόνια και το οποίο χρησιμοποιείται κατά κόρον από τα σεισμολογικά εργαστήρια παγκοσμίως. Αποτελεί μια ολοκληρωμένη σουίτα από επιμέρους εργαλεία για αποθήκευση και επεξεργασία σεισμικών γεγονότων. Ανιχνεύει αυτόματα σεισμούς και υπολογίζει τις συντεταγμένες της εστίας και το μέγεθός τους σε πραγματικό χρόνο. Με τη χρήση γραφικού περιβάλλοντος (GUI), οι δυνατότητες του αυξήθηκαν, κάνοντας πιο εύκολη την οπτικοποίηση και πιο γρήγορη την ανασκόπηση των σεισμών καθώς και τον έλεγχο ποιότητας των αποτελεσμάτων. Ωστόσο αυτή τη στιγμή το λογισμικό δε διαθέτει κάποιο εργαλείο που να υπολογίζει τον ΤΣΡ για τοπικά ή περιφερειακά δίκτυα.

Το ISOLA αποτελεί ένα λογισμικό, γραμμένο κυρίως σε Fortran και MATLAB που υπολογίζει τον ΤΣΡ, με καθοδήγηση από το χρήστη. Ο υπολογισμός του ΤΣΡ βασίζεται στα σεισμολογικά δεδομένα/κυματομορφές καθώς και τις συναρτήσεις Green, τις οποίες το ίδιο το πρόγραμμα υπολογίζει βάσει συγκεκριμένων μεθόδων. Ως προς τον υπολογισμό του ΤΣΡ, η εύρεση της θέσης και του χρόνου που πραγματοποιήθηκε το σεισμικό γεγονός υπολογίζεται μέσα από μια αναζήτηση σε πλέγμα ενώ ο υπολογισμός του ΤΣΡ προκύπτει από διαδικασία αντιστροφής με τη μέθοδο των ελαχίστων τετραγώνων. Ωστόσο, ο υπολογισμός του ΤΣΡ αποτελεί μια επίπονη και χρονοβόρα διαδικασία για το χρήστη, ειδικά σε περιοχές με υψηλή σεισμικότητα όπου ο όγκος των προς ανάλυση δεδομένων είναι μεγάλος.

Όσον αφορά στις βασικές βιβλιοθήκες της Python που χρησιμοποιήθηκαν από το scisola, η βιβλιοθήκη ObsPy, παρέχει εργαλεία αποκλειστικά για την

επεξεργασία σεισμικών δεδομένων. Παρέχει επίσης τη δυνατότητα ανάλυσης (parsing) αρχείων γνωστού τύπου, πρόσβαση σε σεισμολογικά κέντρα καθώς και ρουτίνες επεξεργασίας σημάτων για το χειρισμό των σεισμικών δεδομένων/χρονοσειρών. Η βιβλιοθήκη matplotlib παρέχει τη δυνατότητα δημιουργίας απλών αλλά και πιο πολύπλοκων γραφικών παραστάσεων, όπως αναπαράσταση χρονοσειρών, ιστογράμματα, φάσματα ισχύος κτλ. σε διάφορες μορφές. Επίσης σε συνδυασμό με τη Numpy μπορεί να χρησιμοποιηθεί για μαθηματικούς υπολογισμούς. Η PyQt, για το σχεδιασμό του γραφικού περιβάλλοντος, κάνει χρήση του εργαλείου Qt το οποίο χρησιμοποιείται ευρέως στην ανάπτυξη λογισμικών με γραφικό περιβάλλον. Η βιβλιοθήκη subprocess για να μπορέσει να πραγματοποιηθεί η διασύνδεση με τα διάφορα εργαλεία στο επίπεδο της python γλώσσας προγραμματισμού. Η βιβλιοθήκη, multiprocessing, χάρη στην οποία ο υπολογισμός των συναρτήσεων του Green, αλλά και η αντιστροφή πραγματοποιείται παράλληλα. Και τέλος, οι βιβλιοθήκες MySQLdb και pycpg2 που παρέχουν ρουτίνες για τη διασύνδεση της Python και της MySQL ή της PostgreSQL αντίστοιχα.

Περιγραφή του λογισμικού

Το scisola χρησιμοποιεί τη βάση δεδομένων (MySQL) για να:

- αποθηκεύσει τις πληροφορίες των σταθμών τις οποίες λαμβάνει από το SeisComP3, δίνοντας παράλληλα τη δυνατότητα στο χρήστη να τις τροποποιήσει
- αποθηκεύσει τα αποτελέσματα από τις λύσεις των ΤΣΡ
- αποθηκεύσει τις εκτενείς παραμετροποιήσεις/ρυθμίσεις του scisola

Ο κώδικας του scisola χωρίζεται σε δύο επίπεδα. Στο 2ο επίπεδο, υπάρχουν τα πακέτα lib και gui που είναι υπεύθυνα για την υλοποίηση της «λογικής» του scisola, όπως τους διάφορους αλγορίθμους που χρησιμοποιεί καθώς και τη δημιουργία του γραφικού περιβάλλοντος αντίστοιχα. Στο 1ο επίπεδο που

είναι και το πιο υψηλό, υπάρχει το πακέτο scisola το οποίο ουσιαστικά αποτελεί την ένωση των άλλων δύο πακέτων -lib και gui-.

Η ειδοποίηση για τη γένεση ενός σεισμού, οι πληροφορίες των σεισμολογικών σταθμών καθώς και οι αντίστοιχες κυματομορφές λαμβάνονται από το SeisComP3 μέσω διάφορων εργαλείων και υπηρεσιών όπως το slinktool και seedlink server. Ο ΤΣΡ υπολογίζεται χρησιμοποιώντας το λογισμικό ISOLA και συγκεκριμένα, ο υπολογισμός των συναρτήσεων του Green αλλά και η αντιστροφή πραγματοποιείται παράλληλα μέσω της βιβλιοθήκης multiprocessing της python έτσι ώστε να επιτύχει πιο γρήγορα αποτελέσματα. Οι λύσεις από τον υπολογισμό του ΤΣΡ, αποθηκεύονται στη βάση δεδομένων του scisola για καλύτερη διαχείριση των αποτελεσμάτων. Επιπλέον, το scisola επιτρέπει την εκτενή παραμετροποίησή του με βάση τις ανάγκες του κάθε χρήστη, μέσω γραφικού περιβάλλοντος. Πέρα από την αυτόματη λειτουργία, το scisola, μπορεί να εκτελέσει την αυτόματη διαδικασία με χειροκίνητο τρόπο μέσω python scripts, προκειμένου ο χρήστης να δοκιμάσει διάφορα σετ σεισμών και ρυθμίσεις. Τέλος υποστηρίζει γραφικό περιβάλλον, με το οποίο ο χρήστης μπορεί να πραγματοποιήσει ανασκόπηση των αποτελεσμάτων αλλά ακόμα και να τροποποιήσει τις λύσεις μερικώς, μέσω διάφορων ρυθμίσεων.

Ανάλυση των αλγορίθμων

Η αυτόματη διαδικασία του υπολογισμού του ΤΣΡ μπορεί να περιγραφεί από τα ακόλουθα οκτώ βήματα:

1. Πυροδότηση του γεγονότος
2. Επιλογή σεισμολογικών σταθμών με βάση την απόσταση από το επίκεντρο
3. Φιλτράρισμα προβληματικών σεισμικών δεδομένων
4. Προ-επεξεργασία των σεισμικών δεδομένων
5. Επιλογή σταθμών με βάση την αζιμουθιακή κάλυψη από το επίκεντρο
6. Υπολογισμός των συναρτήσεων του Green
7. Αντιστροφή του ΤΣΡ

8. Δημιουργία διαγραμμάτων με τα αποτελέσματα

Η πυροδότηση του γεγονότος μπορεί να πραγματοποιηθεί είτε με παρακολούθηση σε πραγματικό χρόνο του συστήματος SeisComP3, είτε με χειροκίνητο τρόπο μέσω python scripts.

Η επιλογή των σταθμών βάση της απόστασης πραγματοποιείται με χρήση κανόνων που ορίζει ο χρήστης στις ρυθμίσεις του scisola. Συγκεκριμένα, μπορεί να επιλέξει σταθμούς με βάση την επικεντρική τους απόσταση και το μέγεθος του σεισμού.

Στη διαδικασία του φιλτραρίσματος αφαιρούνται σταθμοί από τη διαδικασία του υπολογισμού, των οποίων οι κυματομορφές είναι ψαλιδισμένες ή περιέχουν κενά καθώς και σταθμοί που τα δεδομένα τους δεν είναι διαθέσιμα από το Seedlink server.

Ακολούθως, πραγματοποιείται η διαδικασία της προ-επεξεργασίας των σεισμικών δεδομένων: αφαίρεση της επίδρασης των οργάνων (deconvolution), απόσπαση σήματος συγκεκριμένης χρονικής διάρκειας και επαναδειγματοληψία (resampling).

Στη συνέχεια, πραγματοποιείται η επιλογή των σταθμών βάση της αζιμουθιακής κάλυψης σε σχέση με το επίκεντρο. Αυτό επιτυγχάνεται χωρίζοντας ένα νοητό κύκλο με κέντρο το επίκεντρο, σε 8 τομείς με κάθε τμήμα να είναι 45° , ο χρήστης επιλέγει πόσοι σταθμοί θα επιλεγούν από κάθε τομέα.

Ο Υπολογισμός των συναρτήσεων του Green πραγματοποιείται από το λογισμικό ISOLA.

Εν συνεχεία, ακολουθεί ο υπολογισμός της αντιστροφής, ο οποίος πραγματοποιείται σε πλέγμα τόσο στο χρόνο όσο και στο χώρο (βάθος) προκειμένου να βρεθεί η βέλτιστη θέση και ο χρόνος του κεντροειδούς (centroid). Η θέση και η χρονική στιγμή με την καλύτερη συσχέτιση επιλέγεται αυτόματα από το ISOLA ως τελική λύση.

Τέλος, πραγματοποιείται η δημιουργία διαγραμμάτων με τα τελικά αποτελέσματα. Στα διαγράμματα περιλαμβάνεται ένας χάρτης με το μηχανισμό γένεσης (TSP) και τους σταθμούς που συνέβαλαν στην επίλυση.

καθώς επίσης και ένα διάγραμμα με τους βέλτιστους σε κάθε χρονική στιγμή αναζήτησης μηχανισμούς γένεσης για κάθε βάθος του πλέγματος, αλλά και ένα συνολικό διάγραμμα με όλους τους μηχανισμούς γένεσης για κάθε βάθος και για κάθε χρονική στιγμή του πλέγματος. Επιπλέον περιλαμβάνεται διάγραμμα με τα πραγματικά και τα συνθετικά δεδομένα των σταθμών που συνέβαλαν καθώς και διάγραμμα με τις πραγματικές κυματομορφές τους, όπως επίσης και ένα αρχείο κειμένου με τα αποτελέσματα.

Κατά τη διαδικασία τροποποίησης/αναθεώρησης των αποτελεσμάτων, ο χρήστης δύναται να αλλάξει τις συχνότητες της αντιστροφής αλλά και να αφαιρέσει σταθμούς που επελέγησαν από την αυτόματη διαδικασία έτσι ώστε να υπολογίσει μια αναθεωρημένη έκδοση του ΤΣΡ.

Ο watcher αποτελεί μια βιβλιοθήκη του scisola η οποία είναι υπεύθυνη στο να «ακούει» το SeisComP3 για ύπαρξη νέου σεισμού. Ο watcher, αρχικά ελέγχει αν το SeisComP3 κατέγραψε ένα νέο σεισμό και ακολούθως λαμβάνει τις πληροφορίες για αυτό το σεισμό, όπως το γεωγραφικό μήκος και πλάτος καθώς και άλλες. Εν συνεχεία, ελέγχει αν είναι εντός συγκεκριμένων ορίων γεωγραφικής περιοχής και μεγέθους σεισμού και τέλος εκκινεί την αυτόματη διαδικασία υπολογισμού του ΤΣΡ, ενώ παράλληλα περιμένει για ένα νέο ενδεχόμενο σεισμικό γεγονός.

Παράδειγμα εφαρμογής

Προκειμένου να αξιολογηθούν οι λύσεις του ΤΣΡ που παράγονται από το scisola, μελετήθηκε ένα σετ από 46 σεισμούς οι οποίοι συνέβησαν από τις 1 Φεβρουαρίου 2014 έως τις 25 Ιουνίου 2014 στην Ελλάδα. Η αξιολόγηση έχει πραγματοποιηθεί συγκρίνοντας τις λύσεις του αυτόματου ΤΣΡ του scisola με τις αντίστοιχες λύσεις του ΤΣΡ όπως υπολογίστηκαν με χειροκίνητο τρόπο από το Γεωδυναμικό Ινστιτούτο του Εθνικού Αστεροσκοπείου Αθηνών -GI NOA-). Για τη σύγκριση χρησιμοποιήθηκε η μετρική Kagan η οποία

προσδιορίζει κατά πόσο διαφέρουν δύο ΤΣΡ. Τα αποτελέσματα είχαν ως εξής:

- 34 στις 46 λύσεις του ΤΣΡ είχαν παρόμοια αποτελέσματα μεταξύ της αυτόματης και της χειροκίνητης λύσης, και επομένως ποσοστό επιτυχίας: 74%
- 39 στις 46 λύσεις του ΤΣΡ είχαν παρόμοια αποτελέσματα όσον αφορά το μέγεθος του σεισμού (Moment magnitude -Mw-) μεταξύ της αυτόματης και της χειροκίνητης λύσης, και επομένως ποσοστό επιτυχίας: 85%. Επιπλέον σε καμία περίπτωση δεν υπήρξε διαφορά πάνω από 0.3 μονάδες του Mw μεταξύ της αυτόματης και της χειροκίνητης λύσης.
- 35 στις 46 λύσεις του ΤΣΡ είχαν παρόμοια αποτελέσματα όσον αφορά το βάθος του σεισμού (seismic source) μεταξύ της αυτόματης και της χειροκίνητης λύσης, και επομένως ποσοστό επιτυχίας: 76%.

Ένα πολύ σημαντικό αποτέλεσμα, αποτελεί το γεγονός πως το scisola μπορεί να αναγνωρίσει το μέγεθος και το βάθος της σεισμικής πηγής με επαρκή ακρίβεια λίγα λεπτά μετά το συμβάν, δεδομένου ότι η αυτόματη διαδικασία διαρκεί συνολικά, περίπου 5 λεπτά. Αυτό είναι πολύ σημαντικό για την γρήγορη εκτίμηση των εδαφικών κινήσεων ή αναφορικά με τον κίνδυνο από τσουνάμι.

Επίσης, να σημειωθεί ότι ο χρήστης δύναται να τροποποιήσει τη λύση μερικώς, μέσα σε λίγα λεπτά, και έτσι με το scisola να υπολογίσει εξαιρετικά ακριβείς λύσεις.

Συμπεράσματα

Οι Τανυστές Σεισμικής Ροπής, όπως προαναφέρθηκε, χρησιμοποιούνται σε ένα ευρύ φάσμα σεισμολογικών ερευνητικών τομέων, όπως η παραγωγή χαρτών εδαφικών κινήσεων (shakemap), προειδοποιήσεις για τσουνάμι, αξιολόγηση της εδαφικής κίνησης κ.ά. Μέχρι τώρα, η διαδικασία υπολογισμού του ΤΣΡ, καθώς και η διάδοση των αποτελεσμάτων, αποτελούσαν μια χρονοβόρα και επίπονη εργασία για τα περισσότερα

περιφερειακά ή τοπικά δίκτυα. Έτσι, ο σκοπός της παρούσας μεταπτυχιακής εργασίας ήταν να υλοποιήσει ένα λογισμικό που να μπορεί να παρέχει αυτόματα λύσεις του ΤΣΡ σεισμών που καταγράφονται στο λογισμικό SeisComP3 σε πραγματικό χρόνο. Η δημιουργία αυτού του λογισμικού -με την ονομασία scisola- αποτελεί την αρχή για την άμεση εφαρμογή αυτοματοποιημένης διαδικασίας υπολογισμού του ΤΣΡ. Επίσης, παρουσιάζει ερευνητικό ενδιαφέρον τόσο στην επιστήμη της σεισμολογίας, όσο και στον κλάδο της επιστήμης των υπολογιστών.

Το λογισμικό (scisola) που υλοποιήθηκε στο πλαίσιο αυτής της μεταπτυχιακής διπλωματικής εργασίας παρέχει τη δυνατότητα αυτοματοποίησης του υπολογισμού του Τανυστή Σεισμικής Ροπής με διασύνδεση στο ευρέως γνωστό λογισμικό, SeisComP3 καθώς επίσης και την αποθήκευση των αποτελεσμάτων σε βάση δεδομένων. Επιπλέον παρέχει τη δυνατότητα παραμετροποίησης του λογισμικού από το χρήστη με χρήση γραφικού περιβάλλοντος (configuration). Τέλος, παρέχει τη δυνατότητα προβολής των αποτελεσμάτων (overview) καθώς επίσης και της τροποποίησης των αποτελεσμάτων βάση επιλογής του χρήστη με χρήση γραφικού περιβάλλοντος (revision).

Table of Contents

1	Introduction.....	1
1.1	Moment Tensor.....	1
1.2	Importance of Moment Tensor calculation.....	2
1.3	International Approach.....	3
1.4	Aims of the master thesis.....	4
1.5	Contribution of the master thesis.....	5
1.6	Methodology of the master thesis.....	6
1.7	Structure of the master thesis.....	7
2	System tools.....	8
2.1	Main software tools and packages.....	8
2.2	SeisComp3.....	8
2.3	ISOLA.....	12
2.4	MySQL.....	14
2.5	Python.....	15
2.6	Python packages.....	16
2.6.1	ObsPy.....	16
2.6.2	Matplotlib.....	17
2.6.3	Numpy.....	18
2.6.4	Subprocess.....	19
2.6.5	Multiprocessing.....	19
2.6.6	PyQt4.....	20
2.6.7	MySQLdb and psycpg2.....	21
3	Description of the software.....	23
3.1	Functionality.....	23
3.2	Architecture analysis.....	24
3.2.1	Database.....	24
3.2.2	Structure.....	28
3.2.3	Interconnection with SeisComp3.....	34
3.3	Algorithm analysis.....	35
3.3.1	Automatic mode.....	35
3.3.2	Revise mode.....	42
3.3.3	Watcher.....	42
4	Case Study.....	45
4.1	Evaluation dataset.....	45
4.2	Scisola settings for the evaluation.....	47
4.3	Results.....	48
4.4	Outcome.....	50
5	User Guide.....	51

5.1	<i>Downloading.....</i>	<i>51</i>
5.2	<i>Usage.....</i>	<i>51</i>
5.3	<i>Installation.....</i>	<i>51</i>
5.3.1	Python.....	51
5.3.2	ISOLA.....	52
5.3.3	Database.....	52
5.4	<i>Execution.....</i>	<i>53</i>
5.5	<i>Browsing.....</i>	<i>53</i>
5.6	<i>Configuration.....</i>	<i>60</i>
6	<i>Conclusion.....</i>	<i>67</i>
6.1	<i>Summary of the master thesis.....</i>	<i>67</i>
6.2	<i>Conclusion.....</i>	<i>67</i>
6.3	<i>Future Improvements.....</i>	<i>68</i>
	<i>Bibliography.....</i>	<i>69</i>

List of Figures

<i>Figure 1.</i> The nine generalized couples of the seismic moment tensor.....	1
<i>Figure 2.</i> Types of beachball plot associated with different fault.[4].....	2
<i>Figure 3.</i> SeisComP3's modules and their description.....	11
<i>Figure 4.</i> Representation of a distributed SeisComP3 system.[10].....	11
<i>Figure 5.</i> Screenshot of the ISOLA-GUI tool for data pre-processing.[12].....	13
<i>Figure 6.</i> Plot example of matplotlib python package.[18].....	18
<i>Figure 7.</i> Search screen example of PyQt usage on linux Mint OS.....	21
<i>Figure 8.</i> Scisola's database tables and attributes.....	26
<i>Figure 9</i> Scisola's database schema.....	27
<i>Figure 10.</i> Scisola's structure schema.....	29
<i>Figure 11.</i> Scisola's structure layering.....	30
<i>Figure 12.</i> Scisola's python code structure.....	33
<i>Figure 13.</i> Scisola's interconnection with SeisComP3.....	34
<i>Figure 14.</i> The 8 steps of the automatic procedure.....	35
<i>Figure 15.</i> Azimuthal distribution.....	38
<i>Figure 16.</i> The generated map containing focal mechanism and stations used.....	40
<i>Figure 17.</i> Plot containing the best inversion results for each depth.....	41
<i>Figure 18.</i> Plot of observed and synthetic waveforms.....	41
<i>Figure 19.</i> Watcher's process triggering in parallel.....	43
<i>Figure 20.</i> The steps of the watcher's functionality.....	44
<i>Figure 21.</i> Distribution of evaluation dataset (red dots are epicenters).....	45
<i>Figure 22.</i> Automatic and manual focal mechanisms of the evaluation dataset.....	46
<i>Figure 23.</i> Scisola's settings for the evaluation dataset.....	48
<i>Figure 24.</i> Kagan value histogram.....	48
<i>Figure 25.</i> Mw difference histogram.....	49
<i>Figure 26.</i> Centroid depth (CD) difference histogram.....	49
<i>Figure 27.</i> Scisola's usage files.....	51
<i>Figure 28.</i> Database log-in.....	53
<i>Figure 29.</i> Main screen.....	54
<i>Figure 30.</i> Main buttons.....	54
<i>Figure 31.</i> Log screen.....	55
<i>Figure 32.</i> Review screen 1.....	56
<i>Figure 33.</i> Review screen 2.....	56
<i>Figure 34.</i> Review screen 3.....	57
<i>Figure 35.</i> Review screen 4.....	57
<i>Figure 36.</i> Review screen 5.....	58
<i>Figure 37.</i> Review screen 6.....	58
<i>Figure 38.</i> Review screen 7.....	59
<i>Figure 39.</i> Review screen 8.....	59
<i>Figure 40.</i> Settings screen 1.....	60
<i>Figure 41.</i> Settings screen 2.....	61
<i>Figure 42.</i> Settings screen 3.....	61
<i>Figure 43.</i> Settings screen 4.....	62
<i>Figure 44.</i> Stations/streams screen.....	62
<i>Figure 45.</i> Scisola settings.....	66

1 Introduction

1.1 Moment Tensor

The seismic Moment Tensor (MT) is a mathematical representation of the movement on a fault during an earthquake, comprising nine generalized couples, or nine sets of two vectors. The tensor depends on the source strength and fault orientation.[1]

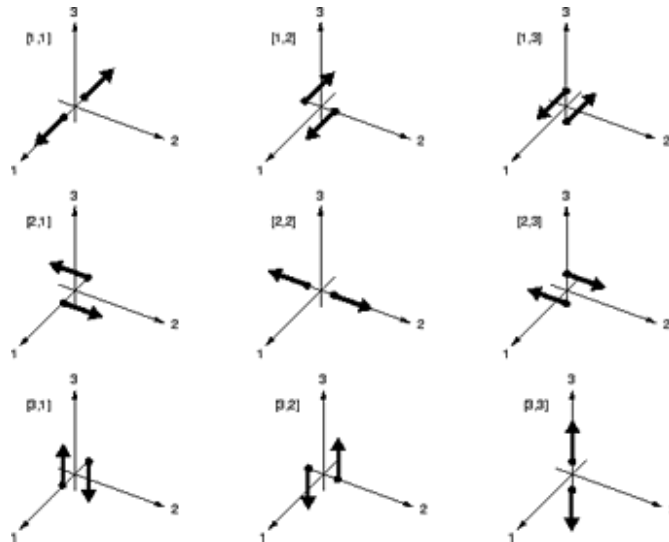


Figure 1. The nine generalized couples of the seismic moment tensor.

Modified after Aki and Richards (1980).[2]

The focal mechanism of an earthquake describes the inelastic deformation in the source region that generates the seismic waves. In the case of a fault-related event it refers to the orientation of the fault plane that slipped and the slip vector and it is also known as a fault-plane solution. Focal mechanisms are derived from a solution of the moment tensor for the earthquake, which itself is estimated by an analysis of observed seismic waveforms. The moment

tensor solution is typically displayed graphically using a so-called beachball diagram.[3]

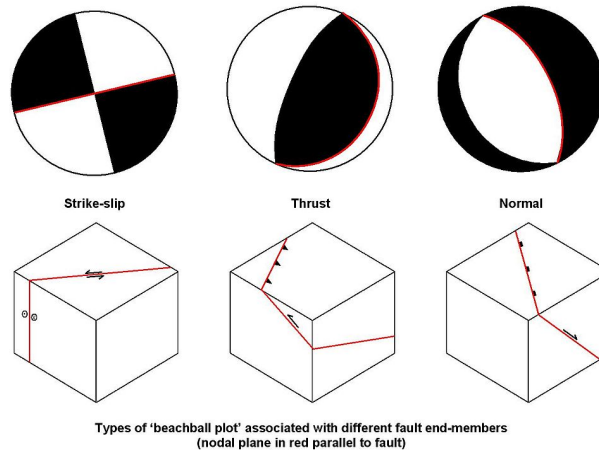


Figure 2. Types of beachball plot associated with different fault.[4]

1.2 Importance of Moment Tensor calculation

The MT is one of the fundamental parameters of earthquakes that can be determined from seismic observations. It is directly related to earthquake fault orientation and rupture direction. The moment magnitude, M_w derived from the moment tensor magnitude, is the most reliable quantity for comparing and measuring the size of an earthquake with other earthquake magnitudes. Moment tensors are used in a wide range of seismological research fields, such as earthquake statistics, earthquake scaling relationships, stress inversion, shakemap generation, tsunami warnings, ground motion evaluation and more.[5]

Fault plane solutions are useful for defining the style of faulting in seismogenic volumes of depth for which no surface expression of the fault plane exists, or where the fault trace is covered by an ocean. Fault plane solutions played the key role in the discovery that the deep earthquake zones in some subducting slabs are under compression and others are under tension.[3]

The procedure of MT calculation as well as the representation and the distribution of its results constitute a manual and a time consuming operation for most local or regional networks. Since MT inversion procedures can provide important real time information for shakemap generation, tsunami warnings, ground motion evaluation and many other studies, their reliable and rapid automation is imperative. Modern seismic networks with broadband sensors and real time digital telemetry provide the necessary information for these types of calculations. Automatic MT's are now provided by global networks and a few very dense regional networks, within minutes after a significant event. [6][7][8]

However, the wide distribution of SeisComP3 (<http://www.seiscomp3.org/>) -a famous software package for seismological purposes- and the effectiveness of ISOLA (<http://seismo.geology.upatras.gr/isola/>) -a moment tensor retrieval software package for manual calculations- made the automation of the MT calculation for SeisComP3, possible. This master thesis presents the scisola; an open-source python based software for automatic MT calculation of seismic events provided by SeisComP3 in real-time.

1.3 International Approach

The calculation of MT procedure can be applied in two network categories:

1. global networks
2. regional or local networks

As regards the first network category, Global CMT (<http://www.globalcmt.org>) (Ekström et al., 2012) already supports a global AMT procedure for strong earthquakes for a few decades. Automatic solutions for global networks are provided by GFZ German Research Centre for Geosciences (<http://www.gfz-potsdam.de/en/home/>) (Saul et al., 2011) as well.

As regards the second network category, Berkeley Seismological Laboratory of University of California (<http://seismo.berkeley.edu/>), since 1993 developed a software (Dreger, 2002) that calculates AMT for regional networks and

earthquakes larger than 3.5 Mw. It has also been used at the Japan National Research Institute for Earth Science and Disaster Prevention (<http://www.bosai.go.jp/e>) and by independent researchers along USA, Europe and Asia. This distribution with different variations has been used by the Mediterranean Network (MedNet) of Italy (<http://mednet.rm.ingv.it>). Services that have performed respective efforts are the Swiss Seismological Service (SED) (<http://www.seismo.ethz.ch>) and the Earthquake and Volcano Information Center of Japan (<http://www.eic.eri.u-tokyo.ac.jp/index-e.html>); the last one applies a different approach. The creation of the Hellenic Unified Seismic Network (HUSN) provided the opportunity to apply an automated MT procedure using the available broad band data from almost one hundred stations. Thus the ISOLA code (Sokos and Zahradník, 2008) was extended towards automatic operation. [6] Specifically, the ISOLA software was converted for automatic use under name amt (Triantafyllis et al., 2013) with the use of Linux OS bash code and other useful tools like nmxptool (Quintiliani, 2007). This initial implementation paved the way for an automatic moment tensor calculation procedure tightly connected with a widely distributed automatic processing software, SeisComP3.[8]

1.4 Aims of the master thesis

The goal of this master thesis is to implement a software that can automatically provide the MT solution of earthquakes that are processed by SeisComP3 in real-time. The creation of this software -so-named scisola- paves the way for immediate implementation in the field of seismology while presenting a research interest in the field of computer science and computer engineering too.

In the context of this master thesis the following will be examined:

- study of various software tools and packages for acquisition and processing of seismic data,

- design and implementation of an automated system for quick calculation of seismic MTs for SeisComP3 software, based on existing international approach; the tools mentioned above will be presented and used in the implementation,
- provision of reliable MT solutions and storing them in a database,
- granting of extensive configuration
- provision of a Graphical User Interface (GUI) for overview and revision of solutions including a simple and easy way of configuration according to user's needs

1.5 Contribution of the master thesis

This master thesis contributes by providing a software tool -scisola- which is an open-source python based application.

It supports:

- automatic calculation of moment tensors; the seismic event notification, station information and the corresponding waveforms are provided by the SeisComP3 system through different utilities and services like slinktool and seedlink server respectively. The moment tensor is calculated through the ISOLA software in parallel mode using multiple threads through multiprocessing python libraries for much faster calculations,
- result storing in database for better data management,
- extensive configuration changes based on the needs of each researcher, through a user friendly graphical interface,
- a graphical overview of the moment tensor calculation and the corresponding data fit
- moment tensor revision in case the user wishes to alter the automatically suggested result.

Scisola uses many open-source python libraries like ObsPy Beyreuther et al. (2010), matplotlib Hunter (2007), PyQt, MySQLdb and psycopg2 and is available to the scientific community for free. [8]

1.6 Methodology of the master thesis

The research, the design and the development of the scisola software which was implemented under the context of this master thesis, was based on six steps:

1. study of the problem to identify and understand the size of the tasks that need to be processed in order to create a reliable information system,
2. investigation of the existing software tools that can be used, in order to discover the range of their potential, identify their functionality, locate the data that they can handle, find the requirements for customization that may arise, etc,
3. design of the software architecture, based on existing international approach,
4. partial implementation of the system in subsystems,
5. identification of future extensions so that this software can be adopted by the seismological and computer engineering community, in order to be a complete and reliable tool for a quick automated process of MT calculation of earthquakes, easy overview of the solution, revision of the solution and extensive configuration in order to satisfy, as much as possible, the user needs,
6. release of the software source code in order to be used and be modified by users and also a resource for the seismological and computer engineering community to make a further step.

1.7 Structure of the master thesis

Initially, the second chapter presents the various software tools and packages, necessary for scisola to function. Specifically, the potentials of these tools are presented here as well as, the functionality that they fulfill and the data they can handle and more.

The third chapter deals with the general structure of the architecture of the scisola software; its functionality, the database support, the structure of the source code and the interconnection with SeisComp3. It also analyzes the algorithms used in order to automate the MT calculation process, to revise a MT solution and to watch SeisComp3 for new incoming events in real-time.

The forth chapter presents a case study of 46 events for which the Automatic MT solutions were calculated and compared to the manual ones.

The fifth chapter presents the user guide in order to provide an easy way to install, execute and browse the functionality of scisola, while configuring scisola's settings at the same time.

In the final chapter there is a summary of the master thesis, the conclusion of the results and future improvements that can be implemented to the scisola software.

2 System tools

2.1 Main software tools and packages

In the context of this master thesis, the implementation of scisola needed various software tools and packages. The main software tools and packages as well as the programming language that was used in order to create scisola, are the following:

- SeisComP3
- ISOLA
- MySQL
- Python programming language
- Python packages:
 - ObsPy
 - Matplotlib
 - Numpy
 - Subprocess
 - Multiprocessing
 - PyQt4
 - MySQLdb
 - pycopg2

Subsequently, the above software tools and packages will be presented in a more extensive way.

2.2 SeisComP3

The seismological software SeisComP3 (<http://www.seiscomp3.org/>) (the 3rd version of SeisComP) is possibly the most widely distributed software package

for seismological purposes, such as pure acquisition or real-time data exchange over Internet to even a fully featured real-time earthquake monitoring, and it has evolved within the last 10 years.

SeisComP3 provides the following features:

- data acquisition
- data quality control
- data recording
- real-time data exchange
- network status monitoring
- real-time data processing
- issuing event alerts
- waveform archiving
- waveform data distribution
- automatic event detection and location
- interactive event detection and location
- event parameter archiving
- easy access to relevant information about stations, waveforms and recent earthquakes

A SeisComP3 automatic system consists of a set of independent applications each performing a discrete task. The communication between the applications is carried out by a TCP/IP based messaging system. This messaging system is based on the open source toolkit Spread that provides a high performance messaging service across local and wide area networks. At the top of Spread, a mediator, *scmaster*, handles additional requirements of SeisComP3 that are not natively provided by Spread. The messaging system is used for the exchange of meta data (e.g. picks) and configurations to some extent. SeisComP3 adheres to community standards where available, such as QuakeML, on which the data model of SeisComP3 and the database object

schema is based, and SEED. The waveform data acquisition is based on the well established SeedLink data transmission protocol which has been the core of SeisComP from the very beginning and became a popular world standard; it was developed at the GFZ Potsdam as the new ArcLink protocol. SeisComP3 supports MySQL, PostgreSQL and SQLite databases.

The applications in SeisComP3 can be divided into four different groups:

1. data acquisition
2. processing
3. graphical user interfaces (GUIs)
4. utilities

Below is the list of all SeisComP3 modules, the group they belong and a short description of what they do.

<i>Application</i>	<i>Type</i>	<i>Description</i>
Seedlink	data acquisition	providing real-time waveform data
ArcLink	data acquisition	providing archive waveform data
scmaster	processing	handling messaging
scqc	processing	determination of waveform quality parameter
scautopick	processing	automatic picking
scautoloc	processing	automatic event detection and localization
scamp	processing	amplitude calculation
scmag	processing	magnitude calculation
scevent	processing	origin association, best, magnitude selection, best origin selection
scrttv	GUI	real-time waveform monitor
scmv	GUI	map overview showing, actual station status and events
scesv	GUI	summary view of most important event information
scolv	GUI	reviewing and revising origins, manual picking tool
scqcv	GUI	showing station quality status
scmm	GUI	message monitoring
scbulletin	utility	creating bulletins of events from the database
scdb	utility	inserting objects from QuakeML file or messaging into the database
scevtlog	utility	logging the event history

scevtls	utility	listing events for a given time range
scevtstreams	utility	listing all waveform streams used for event detection
scimex	utility	exchange of meta data objects between SeisComP3 systems with filter functionality
scimport	utility	forwarding of meta data objects from one messaging system to another
scm	utility	performance monitor similar to UNIX top
scproclat	utility	logging message history
scvoice	utility	event alert with optional voice output
scxmldump	utility	event dump to QuakeML file from database
sczip	utility	zip implementation of SeisComP3

Figure 3. SeisComP3's modules and their description

Below there is also a representation of how a distributed SeisComP3 system works:

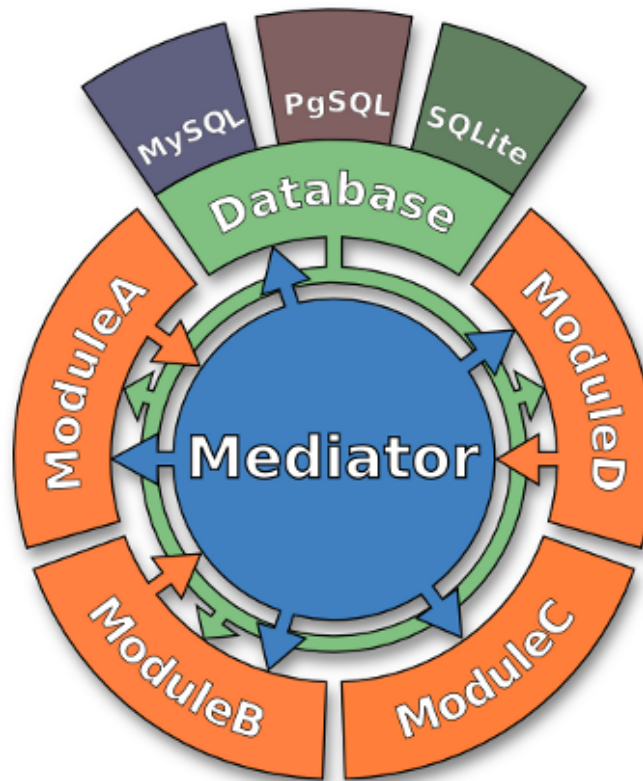


Figure 4. Representation of a distributed SeisComP3 system.[10]

SeisComP3 is a coherent software engineering effort, primarily coded in C++, with most library functionality accessible from Python scripts through a wrapper layer and it runs under Linux (all reasonably recent releases of Ubuntu, SuSE, etc.), Solaris, MacOSX (experimental) while Windows are currently not supported.[9]

2.3 ISOLA

ISOLA (<http://seismo.geology.upatras.gr/isola/>) is a moment tensor retrieval software. It consists of two software packages; ISOLA and ISOLA-GUI.

It allows an easy application of the iterative deconvolution method of Kikuchi and Kanamori (1991), for local and regional events, thus the method can be used for both single and multiple source geometry. Full wavefield is considered, and Green's functions are calculated by the discrete wavenumber method of Bouchon (1981) and Coutant (1989). Moment tensor of subevents is found by least-square minimization of misfit between observed and synthetic waveforms, while position and time of subevents is optimized through grid search. The computational options include inversion to retrieve the full moment tensor (MT), the deviatoric MT, and pure double-couple MT. Finite-extent source inversions may also be performed in the case of a large event.

ISOLA-GUI, is a Graphical User Interface developed in MATLAB with purpose to combine processing speed of the ISOLA Fortran code and user-friendly MATLAB environment. The MATLAB-based GUI facilitates easy data handling, while providing at the same time complete user control during all the processing steps and easy graphical display of the results, using both MATLAB and GMT (GMT- Generic Mapping Tools <http://gmt.soest.hawaii.edu/>) resources. The waveform inversion is done using the Fortran ISOLA code, in order to take advantage of language speed, while user interaction even during the inversion is within the MATLAB environment. Various tools are available for the user in order to explore the data quality, adjust the computational parameters, correct the data, and display inversion results in a publication-ready form. The modular design of the GUI allows the easy upgrade as well as the inclusion of user-created additional modules, using elementary skills in the MATLAB programming language.

ISOLA efficiently combines the Fortran and MATLAB skills. Fortran for the power and speed for compute-intensive wave propagation modeling and source inversion (Green's functions) and MATLAB for quick and user-friendly data handling, control of the inversion process and plotting of the results.

The software was named ISOLA, since the retrieved point-source moment tensors (called subevents) represent isolated asperities, or slip patches of the fault.

The Fortran code ISOLA has been developed since 2003 while the entire source code is documented in a manual, accompanied by a test example and it is available for free.[11][12]

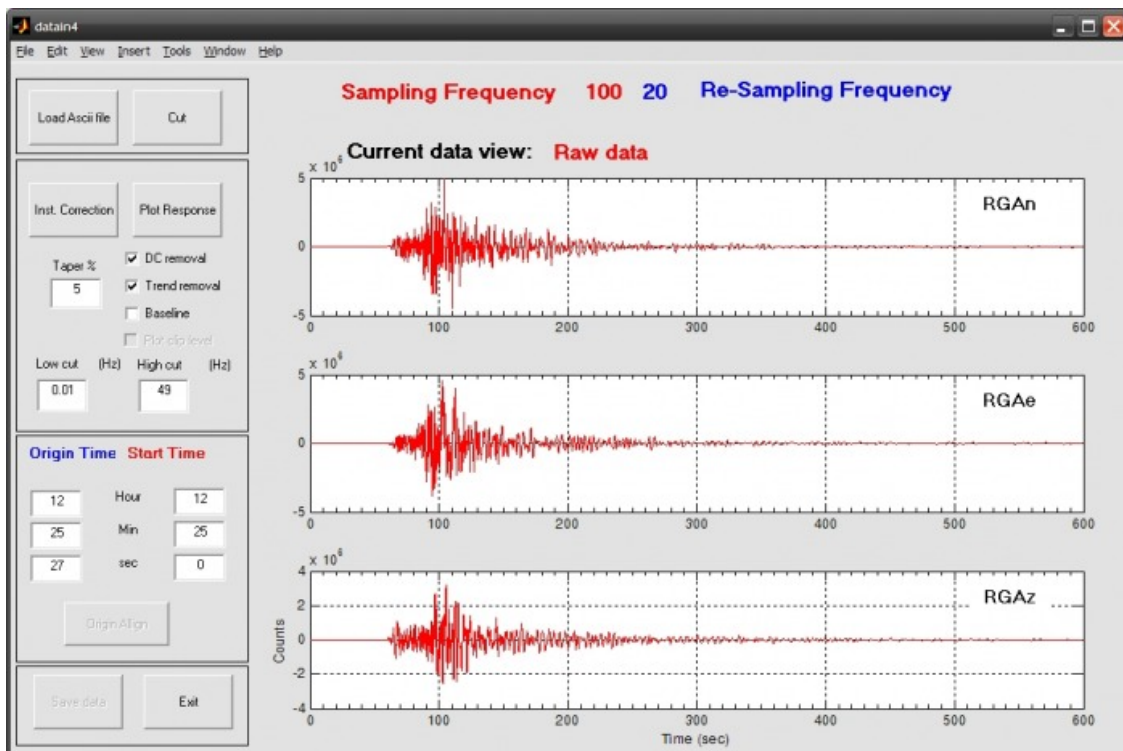


Figure 5. Screenshot of the ISOLA-GUI tool for data pre-processing.[12]

2.4 MySQL

MySQL (<http://www.mysql.com/>) is -as of March 2014- the world's second most widely used open-source relational database management system

(RDBMS). It is a popular database choice for use in web applications, and it is a central component of the widely used LAMP open source web application software stack (and other 'AMP' stacks). MySQL works on many system platforms, including FreeBSD, Linux, OS X, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, and more, while a port of MySQL to OpenVMS also exists. On most Linux distributions the package management system can download and install MySQL with minimal effort, however further configuration is often required to adjust security and optimization settings.

Though MySQL began as a low-end alternative to more powerful proprietary databases, it has gradually evolved to support higher-scale needs as well. It is still most commonly used in small to medium scale single-server deployments, either as a component in a LAMP-based web application or as a standalone database server. Much of MySQL's appeal originates in its relative simplicity and ease of use, which is enabled by open source tools such as phpMyAdmin. In the medium range, MySQL can be scaled by deploying it on more powerful hardware, such as a multi-processor server with gigabytes of memory while it can also be run on cloud computing platforms such as Amazon EC2. There are however limits to how far performance can scale on. It supports a free integrated environment developed by MySQL AB -MySQL Workbench- that enables users to graphically administer MySQL databases and visually design database structures.

MySQL is written in C and C++ while it has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements.[13]

2.5 *Python*

Python (<https://www.python.org/>) is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code

than would be possible in other languages. The language can be used in both small and large scale programs. An empirical study found scripting languages (such as Python) more productive than conventional languages (such as C and Java) for a programming problem involving string manipulation and search in a dictionary. Memory consumption was often "better than Java and not much worse than C or C++". Python supports multiple programming techniques as object-oriented, procedural styles and more. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library. It can also serve for web applications, e.g., via `mod_wsgi` for the Apache web server. Web application frameworks like Django, Pylons, Pyramid, TurboGears, `web2py`, Tornado, Flask and Zope support developers in the design and maintenance of complex applications.

Python has a large standard library, commonly cited as one of Python's greatest strengths, providing tools suited to many tasks. For Internet-facing applications, a large number of standard formats and protocols (such as MIME and HTTP) are supported. Modules for creating graphical user interfaces, connecting to relational databases, pseudorandom number generators, arithmetic with arbitrary precision decimals, manipulating regular expressions, and doing unit testing are also included. Libraries like NumPy, SciPy and Matplotlib allow the effective use of Python in scientific computing, with specialized libraries such as BioPython and Astropy providing domain-specific functionality.

As of January 2014, the Python Package Index, the official repository of third-party software for Python, contains more than 38,000 packages covering a wide range of functionality, including:

- graphical user interface,
- web framework,
- multimedia,
- databases,
- networking and communications,
- test frameworks,

- documentation tools,
- system administration,
- scientific computing,
- text processing,
- image processing

Many operating systems include Python as a standard component; the language ships with most Linux distributions, AmigaOS 4, FreeBSD, NetBSD, OpenBSD and OS X, and can be used from the terminal. A number of Linux distributions use installers written in Python: Ubuntu uses the Ubiquity installer, while Red Hat Linux and Fedora use the Anaconda installer while there is a large number of organizations that make use of Python including Google, Yahoo!, CERN, NASA.

The Python Software Foundation License (PSFL) is a BSD-style, permissive free software license which is compatible with the GNU General Public License (GPL).[14]

2.6 Python packages

Next, the most important packages that were studied and used by scisola, are described.

2.6.1 ObsPy

ObsPy (<https://github.com/obspy/obspy/wiki>) is an open-source project dedicated to providing a Python framework for processing seismological data and simplifies the usage of Python programming for seismologists. It provides parsers for common file formats, clients to access data centers and seismological signal processing routines which allow the manipulation of seismological time series.

In ObsPy the following essential seismological processing routines are implemented and ready to use: reading and writing data SEED/MiniSEED and Dataless SEED, XML-SEED, GSE2 and SAC, as well as filtering,

instrument simulation, triggering, and plotting. There is also support to retrieve data from ArcLink (a distributed data request protocol for accessing archived waveform data) or a SeisHub database. Just recently, modules were added to read SEISAN data files and to retrieve data with the IRIS/FISSURES data handling interface (DHI) protocol. Python gives the user all the features of a full-fledged programming language including a large collection of scientific open-source modules. ObsPy extends Python by providing direct access to the actual time series, allowing the use of powerful numerical array-programming modules like NumPy or SciPy. Results can be visualized using modules such as matplotlib or MayaVi (3D). [15]

ObsPy is under the GNU Lesser General Public License.

2.6.2 Matplotlib

Matplotlib (<http://matplotlib.org/>) is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. It can be used in python scripts, the python and ipython shell (like MATLAB or Mathematica), web application servers and six graphical user interface toolkits. It can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc, with just a few lines of code. [16]

The matplotlib code is conceptually divided into three parts:

- The pylab interface is the set of functions provided by matplotlib. Pylab allows the user to create plots with code quite similar to MATLAB figure generating code.
- The matplotlib frontend or matplotlib API is the set of classes that do the heavy lifting, creating and managing figures, text, lines, plots and so on, which is an abstract interface that knows nothing about output.
- The backends are device-dependent drawing devices (renderers) that transform the frontend representation to hardcopy or a display device. Example backends: PS creates PostScript hardcopy, SVG creates Scalable Vector Graphics hardcopy, Agg creates PNG output using the high

quality Anti-Grain Geometry library that ships with matplotlib, GTK embeds matplotlib in a Gtk+ application, GTKAgg uses the Anti-Grain renderer to create a figure and embed it in a Gtk+ application, and so on for PDF, WxWidgets, Tkinter, etc.

The matplotlib license is based on the Python Software Foundation (PSF) license. [17]

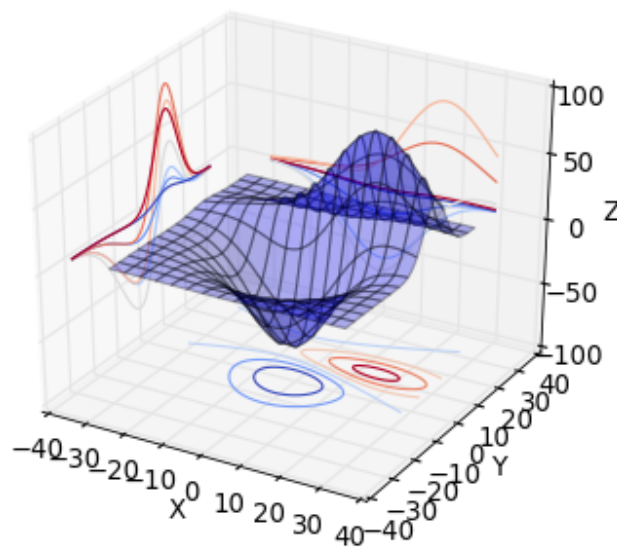


Figure 6. Plot example of matplotlib python package.[18]

2.6.3 Numpy

NumPy (<http://www.numpy.org/>) is the fundamental package for scientific computing with Python. It contains many things such as:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its scientific uses, NumPy can also be used as an efficient multidimensional container of generic data. Arbitrary data-types can be

defined. This allows NumPy to seamlessly and speedily integrate into a wide variety of databases.

Numpy is licensed under the BSD license, enabling reuse with few restrictions. [19]

2.6.4 Subprocess

The subprocess (<https://docs.python.org/2/library/subprocess.html>) module allows you to spawn new processes, connect to their input/output/error pipes, and obtain their return codes. [20]

The subprocess module provides a consistent interface to creating and working with additional processes. It offers a higher-level interface than some of the other available modules. [21] It can be used in order to execute external programs and processes. It can be easily adjusted for python wrapping external programs and processes and use them as python elements. The subprocess module defines one class, Popen and a few wrapper functions that use that class. The constructor for Popen takes arguments to set up the new process so the parent can communicate with it via pipes. [21]

Subprocess is a python's standard library package and is under the Python Software Foundation (PSF) license.

2.6.5 Multiprocessing

Multiprocessing (<https://docs.python.org/2/library/multiprocessing.html>) is a package that supports spawning processes using an API similar to the threading module. The multiprocessing package offers both local and remote concurrency, effectively side-stepping the Global Interpreter Lock by using subprocesses instead of threads. Due to this, the multiprocessing module allows the programmer to fully leverage multiple processors on a given machine. It runs on both Unix and Windows. [22] It can be easily used for parallelizing calculations and processes in order to produce much faster results.

Multiprocessing is a python's standard library package and is under the Python Software Foundation (PSF) license.

2.6.6 PyQt4

PyQt (<http://www.riverbankcomputing.co.uk/software/pyqt/intro>) is a Python binding of the cross-platform GUI toolkit Qt. It is one of Python's options for GUI programming. PyQt is implemented as a Python plug-in. [23]

PyQt brings together the Qt C++ cross-platform application framework and the cross-platform interpreted language Python. Qt includes abstractions of network sockets, threads, Unicode, regular expressions, SQL databases, SVG, OpenGL, XML, a fully functional web browser, a help system, a multimedia framework, as well as a rich collection of GUI widgets. Qt classes employ a signal/slot mechanism for communicating between objects that is type safe but loosely coupled making it easy to create re-usable software components. Qt also includes Qt Designer, a graphical user interface designer where Python code can be generated from it. It is also possible to add new GUI controls written in Python to Qt Designer. A programmer has all the power of Qt, but is able to exploit it with the simplicity of Python.[25]

PyQt implements around 440 classes and over 6,000 functions and methods including:

- a substantial set of GUI widgets
- classes for accessing SQL databases (ODBC, MySQL, PostgreSQL, Oracle)
- QScintilla, Scintilla-based rich text editor widget
- data aware widgets that are automatically populated from a database
- an XML parser
- SVG support
- classes for embedding ActiveX controls on Windows (only in commercial version)[23]

PyQt, provides an ideal combination for rapidly creating powerful and flexible GUI applications. PyQt applications can run on any platform that has PyQt, Python, and the Qt libraries installed, simply by copying the application — and with no source code changes necessary while it is also possible to create stand- alone executables. The currently supported platforms include all versions of Windows from Windows 98 to Vista, and most Unixes including Mac OS X, Linux, Solaris, and HP-UX.[24]

PyQt is available under similar terms to Qt versions older than 4.5; this means a variety of licenses including GNU General Public License (GPL) and commercial license, but not the GNU Lesser General Public License (LGPL). [23]



Figure 7. Search screen example of PyQt usage on linux Mint OS.

2.6.7 MySQLdb and psycopg2

MySQLdb (<http://mysql-python.sourceforge.net/MySQLdb.html>) is a thread-compatible interface to the popular MySQL database server that provides the Python database API. MySQLdb is a thin Python wrapper around `_mysql` which makes it compatible with the Python DB API interface (version 2).[26]

Psycopg2 (<https://pypi.python.org/pypi/psycopg2>) is a PostgreSQL database adapter for the Python programming language. Psycopg2 was written with the aim of being very small, fast and stable. It is different from the other

database adapters because it was designed for heavily multi-threaded applications that create and destroy lots of cursors and make a conspicuous number of concurrent INSERTs or UPDATEs. Psycopg2 also provides full asynchronous operations and supports for coroutine libraries. [27]

Both python packages (MySQLdb and Psycopg2) can be used in order to handle databases, through a nice and easy to use python wrapper.

3 Description of the software

3.1 Functionality

The scisola software is an open-source python based application.

It supports:

1. Automatic Moment Tensor calculation of events provided by SeisComP3 in real-time
2. Easy Moment Tensor solution overview
3. Quick Moment Tensor solution revision
4. Extensive configuration

The seismic events notification, station information and the corresponding waveforms are provided by the SeisComP3 system through different utilities and services like slinktool and seedlink server respectively. The moment tensor is calculated through the ISOLA software in parallel mode using multiple threads through multiprocessing python libraries for much faster calculations. Furthermore, the results of the calculations are saved in a database for a better data management. Scisola allows extensive configuration changes based on the needs of each researcher, through a user friendly graphical interface. Beside the real-time moment tensor operation, it also provides an “offline” mode for testing purposes or calculations. Finally, it supports a graphical review of the moment tensor calculation and the corresponding data fit; as well as the moment tensor revision in case the user wishes to alter the automatically suggested result.[8]

3.2 *Architecture analysis*

3.2.1 *Database*

The scisola software uses a MySQL database. The usage of a well-known database (MySQL) supports a better data management. MySQL database can also be used by external tools through which the user can have a better way of searching through the results or upload the MT results on a web page or even send them via e-mail. This can be achieved through many utilities or programming languages that can connect to MySQL and handle the data, in order to distribute the MT calculations results through the world or create useful statistics.

The scisola database is used for:

- saving and editing station and stream information retrieved from the SeisComP3 software
- saving the earthquake event information and the MT calculation including the streams contributing to the inversion
- saving the extensive configuration of scisola settings

The database consists of ten tables, which can be classified in three categories:

- i. Stations (tables: Station, Stream)
- ii. Settings (tables: Settings, Distance_selection, Inversion_time, Inversion_frequency)
- iii. Event (tables: Event, Origin, Moment_Tensor, Stream_Contribution)

In figure 7 all tables of the scisola database are presented including their description and attributes.

In figure 8 a scisola database schema that has been created according to the MySQL standards is presented including all the tables, their attributes and their connections to each other.

<i>Scisola Database</i>		
<i>Table</i>	<i>Description</i>	<i>Attributes</i>
Station	The Station table is used for manipulating the seismic stations	id, code, network, description, latitude, longitude, elevation, priority
Stream	The Stream table is used for manipulating the available streams for each seismic station, such as HHN, HHE, HHZ, etc	Station_id (foreign key to Station.id), code, azimuth, dip, gain_sensor, gain_datalogger, norm_factor, nzeros, zeros_content, npoles, poles_content, priority
Settings	The Settings table is used for manipulating the scisola configuration	timestamp, center_latitude, center_longitude, distance_range, magnitude_threshold, min_sectors, stations_per_sector, sources, source_step, clipping_threshold, time_grid_start, time_grid_step, time_grid_end, watch_interval, process_delay, process_timeout, crustal_model_path, output_dir, isola_path, sc3_path, sc3_scevtls, sc3_scxmldump, seedlink_path, seedlink_host, seedlink_port
Distance_selection	The Distance_selection table is part of the Settings object and is used for saving the “selection according to distance” rules	min_magnitude, max_magnitude, min_distance, max_distance, Settings_id (foreign key to Settings.id)
Inversion_time	The Inversion_time table is part of the Settings object and is used for saving the “inversion time window (tl)” rules	min_magnitude, max_magnitude, tl, Settings_id (foreign key to Settings.id)
Inversion_frequency	The Inversion_frequency table is part of the Settings object and is used for saving the “inversion frequencies” rules	min_magnitude, max_magnitude, frequency_1, frequency_2, frequency_3, frequency_4, Settings_id (foreign key to Settings.id)

<i>Scisola Database</i>		
<i>Table</i>	<i>Description</i>	<i>Attributes</i>
Origin	The Origin table is used for manipulating the incoming origins (earthquakes)	id, timestamp, description, datetime, magnitude, latitude, longitude, depth, automatic, results_dir
Event	The Event table is used for manipulating the uniqueness of earthquakes. One Event can contain many Origin objects	id, Origin_id (foreign key to Origin.id)
Moment_Tensor	The Moment_Tensor table is part of the Origin object and is used for manipulating the moment tensor solutions for incoming origins (earthquakes). One Origin can contain only one Moment_Tensor object	cent_shift, cent_time, cent_latitude, cent_longitude, cent_depth, correlation, var_reduction, mw, mrr, mtt, mpp, mrt, mrp, mtp, vol, dc, clvd, mo, strike, dip, rake, strike_2, dip_2, rake_2, p_azm, p_plunge, t_azm, t_plunge, b_azm, b_plunge, minSV, maxSV, CN, stVar, fmVar, frequency_1, frequency_2, frequency_3, frequency_4, Origin_id (foreign key to Origin.id)
Stream_contribution	The Stream_contribution table is part of the Origin object and is used for saving the contributing streams' information to the inversion procedure. One Origin can contain many Stream_contribution objects	streamNetworkCode, streamStationCode, streamCode, var_reduction, mseed_path, Origin_id (foreign key to Origin.id)

Figure 8. Scisola's database tables and attributes.

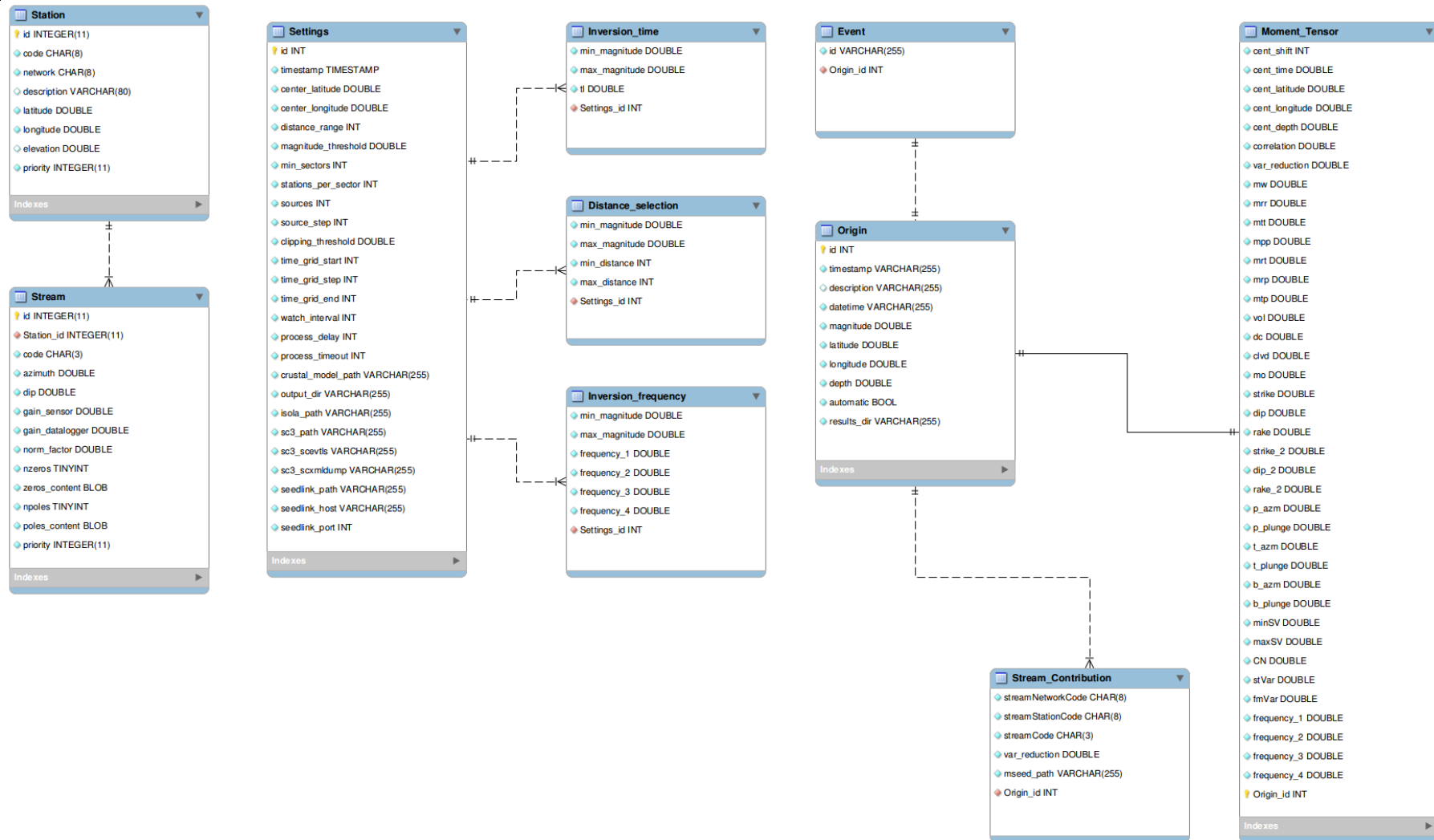


Figure 9 Scisola's database schema.

3.2.2 Structure

The scisola software is an open-source python based application. It uses many python packages interconnected in a special way in order to perform all necessary tasks of scisola. A short description on how scisola uses these main packages follows:

- ObsPy, for seismological purposes such as signal handling, instrumental effect removal and more
- matplotlib, for plots and calculations
- NumPy, for calculations
- PyQt4, for constructing the Graphical User Interface (GUI) such as the review screen, the settings screen and more
- subprocess, for creating a python wrapping to the desired ISOLA and SeisComP3 modules necessary for the scisola software
- multiprocessing, for parallelizing calculations, such as the Green's functions calculation and the inversion procedure in order to produce much faster results
- MySQLdb and psycopg2, which are used for manipulating database queries, handling stations and streams information, saving results and more

In figure 10 a schema containing the main python modules as well as the modules from SeisComp3 and ISOLA that are necessary for scisola in order to run, is presented.

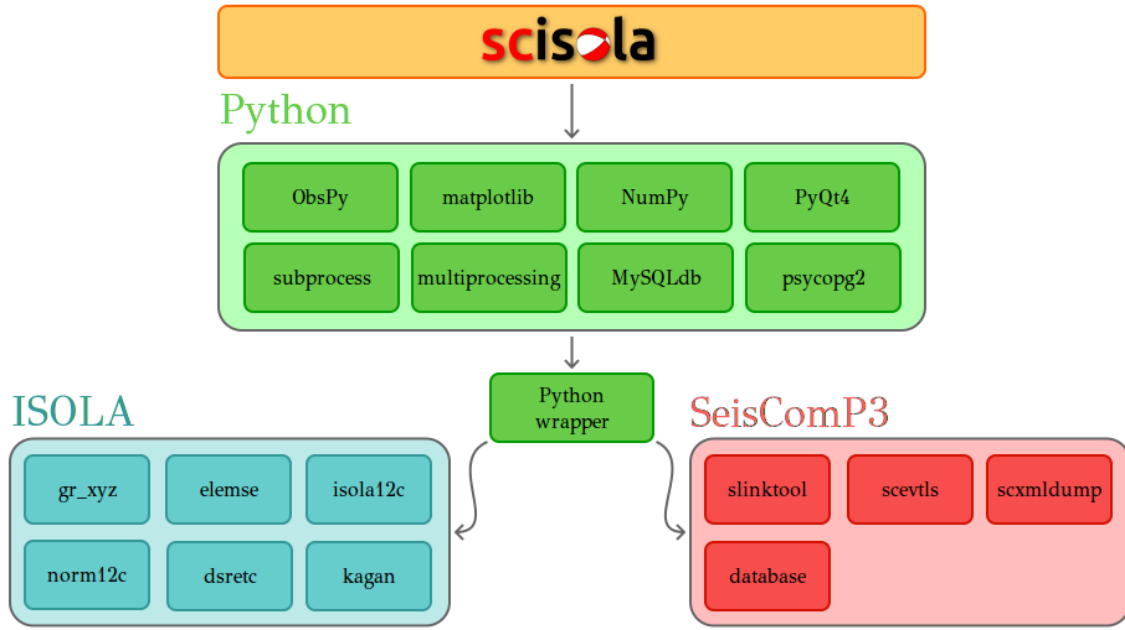


Figure 10. Scisola's structure schema.

As it can be seen in figure 11, scisola consists of three packages in two layers:

- scisola (1st layer -red box-)
- lib (2nd layer -blue box-)
- gui (2nd layer -yellow box-)

The gui package includes all necessary files of the Graphical User Interface (GUI), while the lib package includes all necessary files to implement the logic of scisola which includes all necessary functions and algorithms. The scisola package is the union of the other two packages and it belongs to the 1st layer because it is an abstract layer that hides the whole implementation of the 2nd layer's functionality which includes gui and lib packages. The 1st layers consist of three sub-layers; the lower sub-layer includes all the individual windows while the middle sub-layer includes the main window and the higher sub-layer is just a caller of the middle sub-layer.

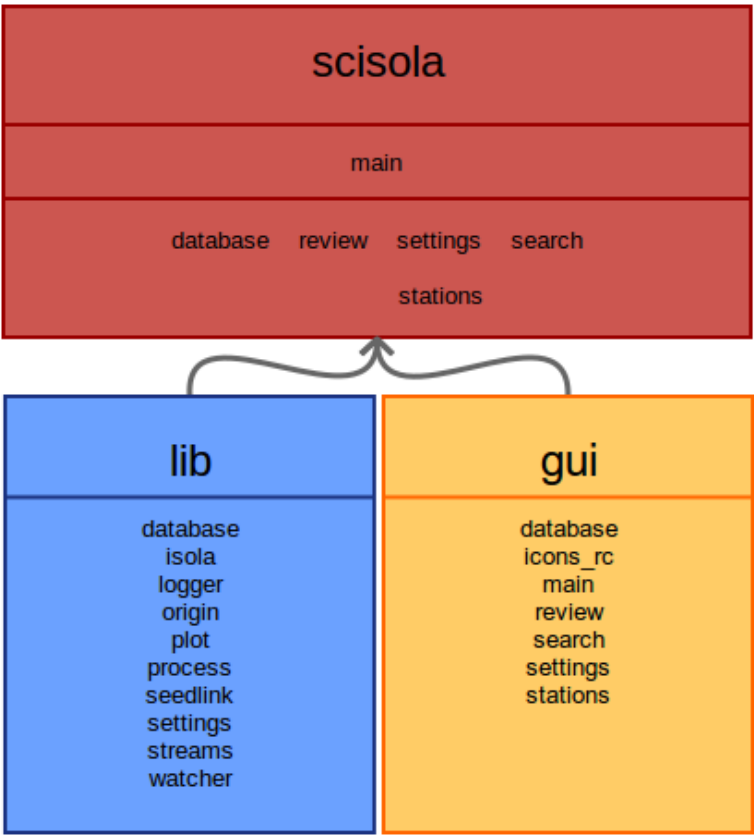


Figure 11. Scisola’s structure layering.

In figure 12 all the individual python files of scisola including their description and the folder they belong to are presented.

<i>Scisola Structure</i>			
<i>Category</i>	<i>Folder</i>	<i>File</i>	<i>Description</i>
-	Scisola	scisola.py	This file is a caller of the main executable file of scisola.py. The user needs to run this executable in order to launch scisola
main	scisola/src	database.py	This is the database window with its full functionality
main	scisola/src	review.py	This file executes the overview window of the MT solutions' results, including the revision panel for the MT revise procedure and contains its full functionality
main	scisola/src	settings.py	This file executes the settings window including its full functionality where the user needs to configure scisola settings or edit existing stations and streams
main	scisola/src	stations.py	This file executes the stations window where the user can edit the existing stations and streams but is being called by the settings.py
main	scisola/src	search.py	This file executes the search windows including its full functionality, where the user can search for previous events that have been calculated according to a datetime range
main	scisola/src	about.py	This file executes the about window containing its full functionality, where the user can read the information of the program such as the version of scisola that runs
main	scisola/src	main.py	This file executes the main executable file of scisola, including its full functionality, that is being called by the scisola.py
lib	scisola/src/lib	database.py	This file contains the logic; the algorithms and the functions in order for scisola to manipulate all the database queries, such as saving MT results or reading stations and streams from SeisComP3
lib	scisola/src/lib	seedlink.py	This file contains the logic; the algorithms and the functions in order for scisola to manipulate all the seedlink connections and queries to SeisComP3, such as retrieving mseed files

<i>Scisola Structure</i>			
<i>Category</i>	<i>Folder</i>	<i>File</i>	<i>Description</i>
lib	scisola/src/lib	isola.py	This file contains the python wrapping in order to run the isola Fortran code through the python api and also the logic; the algorithms and the functions necessary for the inversion procedure
lib	scisola/src/lib	stream.py	This file contains the logic; the algorithms and the functions in order for scisola to manipulate stations/streams and streams data such as data correction by removing instrumental effect
lib	scisola/src/lib	origin.py	This file contains the logic; the algorithms and the functions in order for scisola to manipulate the events, the origins and the MT solutions
lib	scisola/src/lib	plot.py	This file contains the logic; the algorithms and the functions in order for scisola to manipulate the results' plotting of the MT calculations
lib	scisola/src/lib	settings.py	This file contains the logic; the algorithms and the functions in order for scisola to manipulate the configuration of its settings, such as setting the distance “rule” in order scisola to select stations at an incoming event
lib	scisola/src/lib	logger.py	This file contains the logic; the algorithms and the functions in order for scisola to manipulate logging messages for the main logger and the individual loggers for each origin
lib	scisola/src/lib	process.py	This file contains the logic; the algorithms and the functions in order for scisola to run the automatic or revise procedure of the MT calculations of an event
lib	scisola/src/lib	watcher.py	This file contains the logic; the algorithms and the functions in order for scisola to watch SeisComP3 real-time for new events
gui	scisola/src/lib/gui	database.py	This file contains only the Graphical User Interface (GUI) of the database window
gui	scisola/src/lib/gui	review.py	This file contains only the Graphical User Interface (GUI) of the review window

<i>Scisola Structure</i>			
<i>Category</i>	<i>Folder</i>	<i>File</i>	<i>Description</i>
gui	scisola/src/lib/gui	settings.py	This file contains only the Graphical User Interface (GUI) of the settings window
gui	scisola/src/lib/gui	stations.py	This file contains only the Graphical User Interface (GUI) of the stations window
gui	scisola/src/lib/gui	search.py	This file contains only the Graphical User Interface (GUI) of the search window
gui	scisola/src/lib/gui	about.py	This file contains only the Graphical User Interface (GUI) of the about window
gui	scisola/src/lib/gui	icons_rc.py	This file contains the icons that are used by the Graphical User Interface (GUI) of all windows
gui	scisola/src/lib/gui	main.py	This file contains only the Graphical User Interface (GUI) of the main window

Figure 12. Scisola's python code structure.

3.2.3 Interconnection with SeisComP3

The interconnection of scisola with the SeisComP3 software is being implemented by specific python packages. Specifically the scisola packages are:

- database
- seedlink
- watcher

As shown in figure 12, the interconnection of scisola with SeisComP3 is being achieved as follows:

- The database manipulation of SeisComP3 is being implemented by the database package which handles the Stations and Streams information provided by SeisComP3.
- The use of slinktool in order to retrieve the corresponding waveforms is being implemented by the seedlink package.
- The use of scevtls and scxmldump, which are needed in order to watch SeisComP3 for new earthquake events in real-time and for retrieving the corresponding event and origin information respectively, is being implemented by the watcher package.

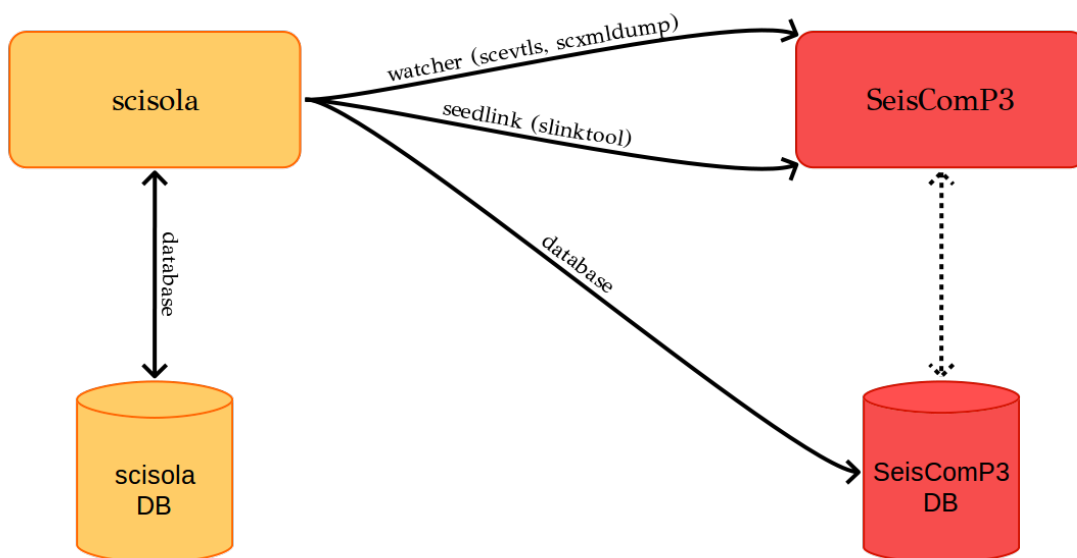


Figure 13. Scisola's interconnection with SeisComP3.

3.3 Algorithm analysis

3.3.1 Automatic mode

The automatic procedure for the MT calculation of an earthquake event, as described in figure 13, can be divided in eight steps:

1. Origin Triggering
2. Station Selection based on Distance
3. Bad Station Data Filtering
4. Station Data Correction
5. Station Selection based on Azimuth
6. Green's Functions Computation
7. Inversion Computation
8. Result Plotting

In figure 14, the description of the eight steps mentioned above is given.

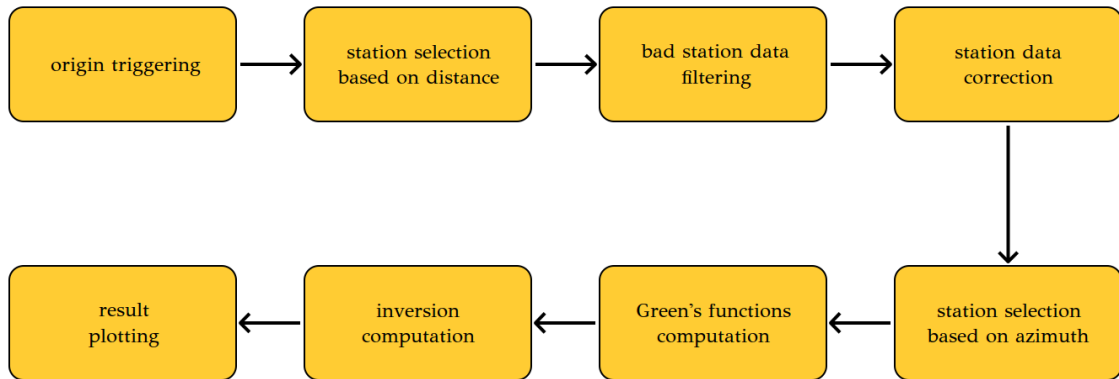


Figure 14. The 8 steps of the automatic procedure.

1. Origin Triggering

Scisola supports two modes in order to run the automated inversion procedure:

- automatic mode
- manual mode

At the first mode, scisola is watching SeisComP3 in real-time by using the watcher python package of scisola.

The watcher is listening to the SeisComP3 software in real-time for new incoming events. This is done by using the scevtls and sxcmlump modules

of SeisComP3 through watcher's python wrapping code. The first module, scevts, is needed for watching SeisComP3 for new earthquake events in real-time while the second module, scxmldump, is needed for retrieving the corresponding event and origin information in order to trigger the automated inversion procedure.

At the second mode, the automated inversion procedure can be executed through python scripting (mostly for testing purposes). This can be achieved by importing the process package of scisola to the corresponding python script and running the automated process with the desired input parameters. The input parameters such as the origin information or the settings for the inversion can be set by the user.

2. Station Selection based on Distance

After the origin triggering, scisola will retrieve the station and their stream information from the scisola database. This information has been added in scisola database by importing it from the SeisComP3 database after the initial scisola configuration. At the next step, scisola will filter streams by certain type (e.g. HHN, HNE) that can be used for the inversion procedure. The accepted types are streams with Band Code: H or B, Instrument Code: H, L or N and Orientation Code: N, E or Z. Afterwards, scisola will remove blacklisted Stations/Streams that are defined by the user in scisola configuration. The blacklisted stations or streams are defined by setting the priority of stations or streams respectively to zero. Thereafter, scisola will calculate the distance and the azimuth of the stations according to the event's epicenter location. Finally, it will choose those stations that are within a distance range; set according to the "distance rules" that have been defined by the user at the scisola configuration. For example, this rule (e.g. $3.5 \leq mw \leq 4.5 \rightarrow 20 \leq \text{distance} \leq 100 \text{ km}$) will choose stations that are within this distance range while the rule will be triggered only if the event is within this magnitude range.

3. Bad Station Data Filtering

After the selection of stations according to the "distance rules", scisola will filter the unavailable streams according to the information from the seedlink server of the SeisComP3 software. Then it will retrieve seismic waveforms in mini-seed format. The date-time range is being set by the date-time of the origin and according to the "time window length (tl) of inversion rules" that have been defined by the user at the scisola configuration. At the next step, it will filter those streams that have gaps or are clipped. The default value of the clipping threshold can be by-passed by the user.

4. Station Data Correction

At this step, scisola will rotate data automatically where necessary according to streams' information on dip and azimuth of the seismometer, as well as the orientation of the stream. For example, if a stream is a "Z" component and has a dip value of 90 degrees, it will reverse its seismic data. If any of the streams contain unacceptable values or combination of values, it will be removed from the process. Afterwards, scisola will apply corrections to the seismic waveforms. The correction step, will initially align the data according to the origin time and cut them to a predefined duration according to the "tl rule" mentioned above; then it will remove the instrumental effect according to poles and zeros streams information and finally, it will re-sample the data according to the "tl rule".

5. Station Selection based on Azimuth

At this step, scisola creates an imaginary circle as in figure 15. This circle is divided into 8 sectors, each 45 degrees wide and at the center of it, is the epicenter. Then the available stations will be distributed based on their azimuth which is calculated according to the epicenter's location. Thereafter, scisola will choose the stations according to the maximum number of stations per sector, which can be defined by the user at the scisola configuration. The user can also set the minimum number of sectors that must contain at least a

station in order to begin the MT calculation. If the stations are well distributed according to the epicenter's location the chances for a better MT solution quality are increased. If the number of stations exceed the value of the maximum number of stations per sector, scisola will choose those stations that have higher priority and are closer to the epicenter's location. By default, the value of stream priority is 7 for high gain seismometers (H), 6 for low gain seismometers (L) and 5 for any other case (e.g. accelerometers).

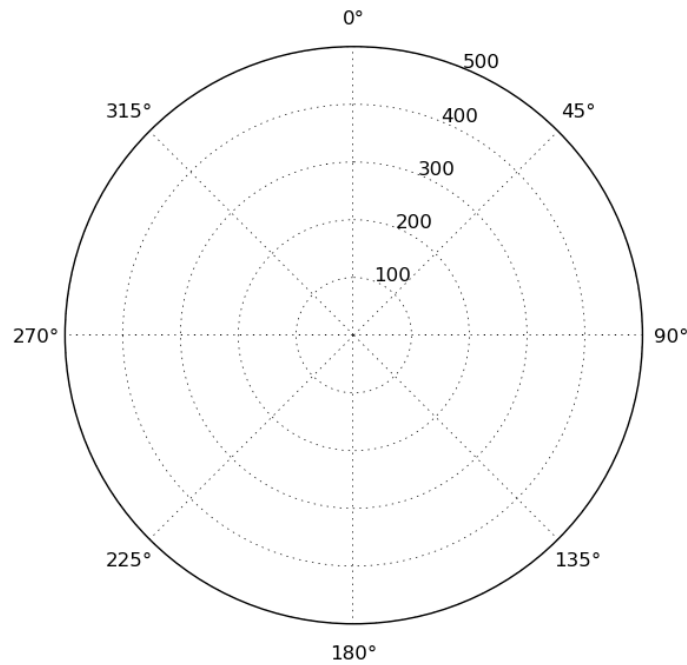


Figure 15. Azimuthal distribution.

6. Green's Functions Computation

After the stations' selection according to the azimuth, scisola will calculate the Green's functions. For the calculation, the code computes the corresponding Green's functions, using a 1D crustal model (which is defined by the user at scisola configuration) and the centroid trial position-station geometry. These are later convolved with a delta time function and six elementary focal mechanisms in order to form elementary seismograms that will be used in the inversion. [6] The centroid horizontal position remains fixed at the epicenter's

location but its depth is grid searched. Starting from the automatic depth estimation, scisola defines a number of trial sources above and below it, where the number is selected by the user at the scisola configuration. The process for each depth is computed in parallel by using multiple threads through the multiprocessing python package, in order to achieve quicker calculations.

For the preparation of the inversion procedure, scisola uses the "time window length (tl) of inversion rules", that have been mentioned above, in order to define the time range of inversion. For example, this rule (e.g. $3.5 \leq mw \leq 4.5 \rightarrow tl = 327.68 \text{ sec}$) will compute the MT inversion using a time window length of 327.68 seconds only if the event takes place within this magnitude range. Finally, in order to run the inversion procedure, there is a limit of 21 stations that can contribute to it.

7. Inversion Computation

As soon as corrected data and elementary seismograms are available the inversion procedure starts. Although the ISOLA code offers various options for source inversion e.g. full moment tensor, deviatoric etc; the deviatoric type is predefined to scisola since it is adequate for most cases of tectonic earthquakes. The inversion frequency band is being set according to the "frequency rules" that have been defined by the user at the scisola configuration. For example, this rule (e.g. $3.5 \leq mw \leq 4.5 \rightarrow \text{frequencies} = [0.04, 0.05, 0.08, 0.09] \text{ Hz}$) will calculate the inversion procedure by using these defined frequencies in the inversion while the rule will be triggered only if the event takes place within this magnitude range. Finally, the centroid time is grid searched a number of seconds before and after the origin time, which is also defined by the user. The inversion procedure for each depth is computed in parallel by using multiple threads through the multiprocessing python package, in order to achieve quicker calculations.

8. *Result Plotting*

After the MT calculation, scisola will generate a text file of the results including the final focal mechanism, which is suitable for distribution over the web. It will also generate a map containing the focal mechanism at the epicenter's location and the location of stations contributing to the inversion.

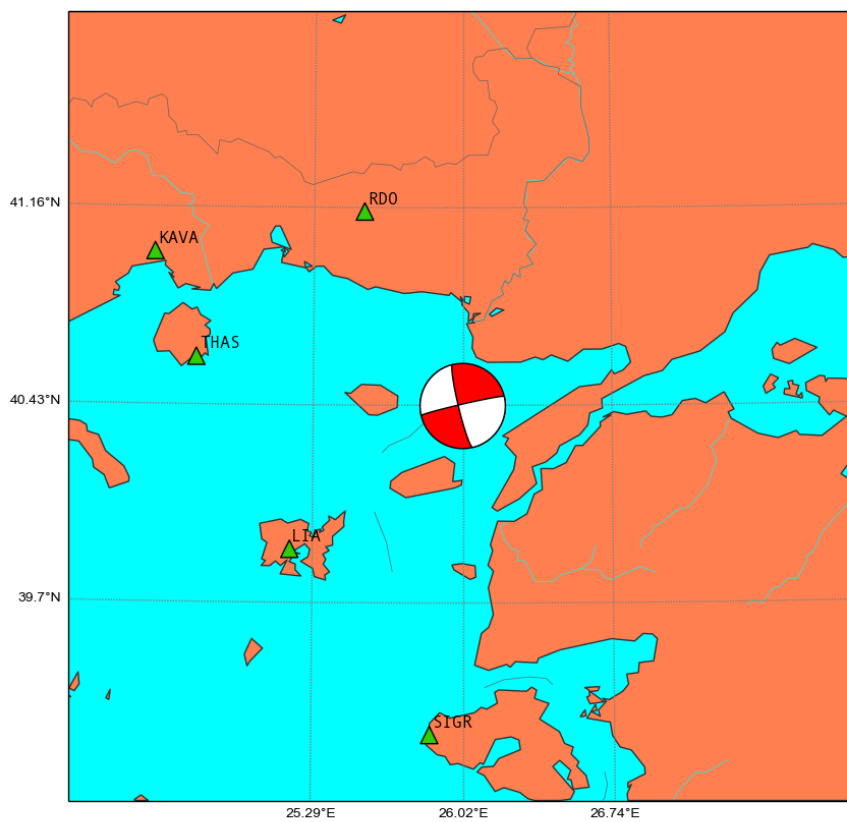


Figure 16. The generated map containing focal mechanism and stations used.

In addition, it will generate a plot of observed and synthetic waveforms. Moreover, scisola will create a plot containing the best focal mechanisms for each depth while it will also create a plot with all the focal mechanisms for each depth and for each time step search.

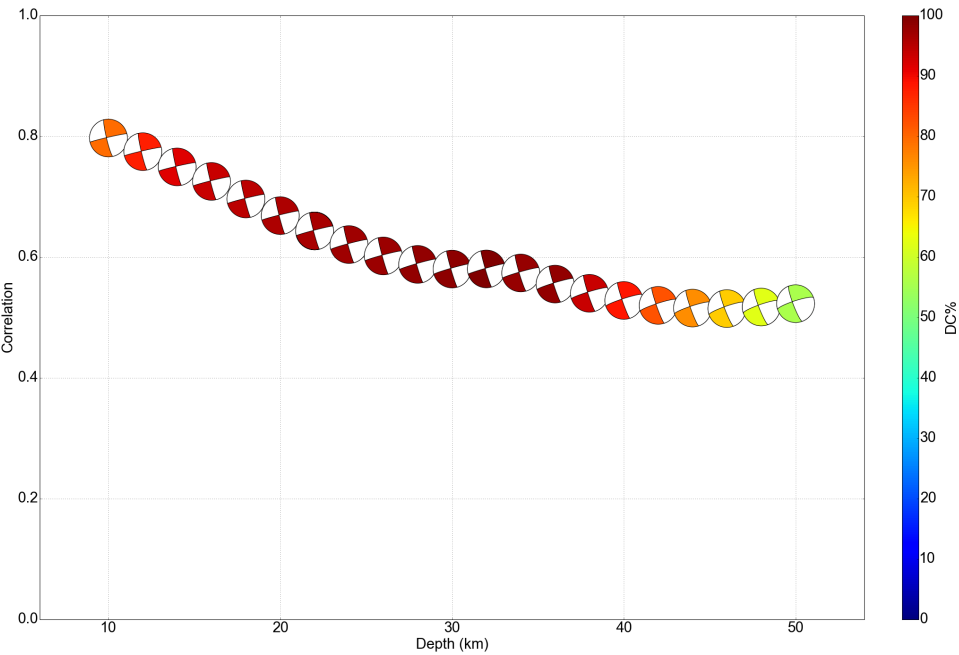


Figure 17. Plot containing the best inversion results for each depth.

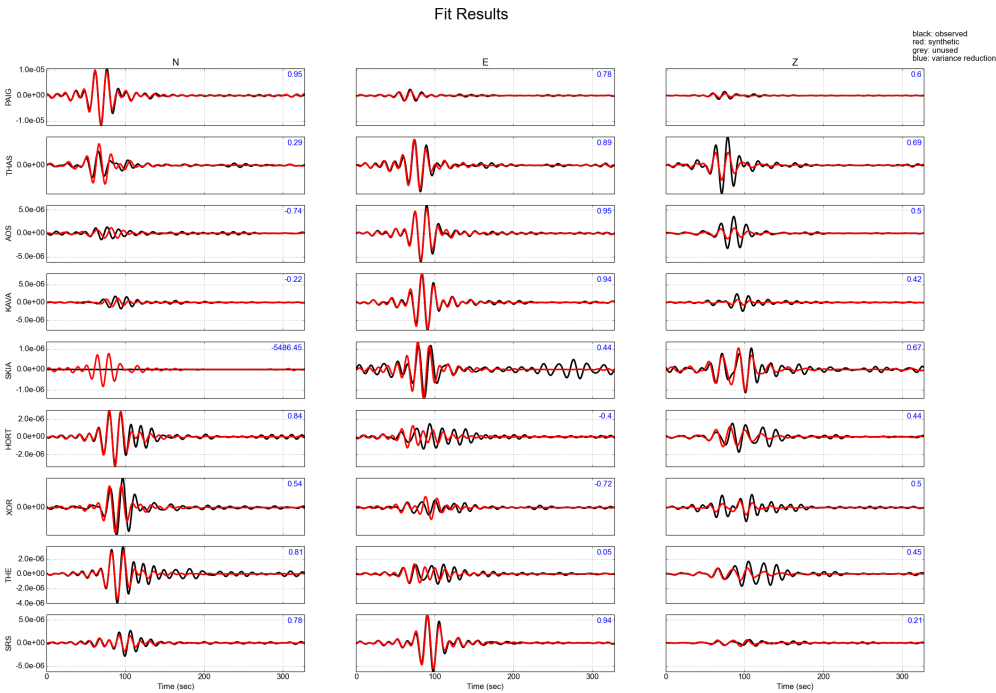


Figure 18. Plot of observed and synthetic waveforms.

Scisola will also generate a plot with the contributing, to the inversion, stations. Finally, the user may select whether to save or not the results to the scisola database.

3.3.2 Revise mode

Scisola supports the possibility of revising the MT solution provided by the automatic procedure, in case the user wishes to alter the automatically suggested options (according to selection based on distance, selection based on azimuth and the various filters). The revise procedure calculations start from the inversion step and end to the plotting results step. Revising can be achieved in two ways:

1. manual removal of streams that have been selected by the automatic procedure, according to user
2. change of the inversion frequencies, according to user. However, the maximum frequency that can be set is defined by the frequency used in the Green's functions computation

3.3.3 Watcher

As shown in figure 19, the watcher package of scisola supports parallel triggering of new earthquake events. In this way, if an event triggered and scisola hasn't finished with its MT calculation, watcher is able to run another scisola process for a new incoming event, in parallel.

The watcher listens to SeisComP3 for a new event arrival within a time range that has been set at the configuration of the scisola by the user. The listening is being implemented by using a python wrapper for the scevtls utility of SeisComP3. If an event id occurs, scisola checks if it is a unique event for the scisola database in order to proceed to the next step. If it is unique, the watcher retrieves the event's information such as date, time, latitude, longitude, magnitude and depth. The information retrieval is implemented by using a python wrapper for the scxmldump utility of SeisComP3. At the next step, the watcher checks if the event is within the magnitude and location

threshold that have been set at the configuration of the scisola settings. Finally, at the last step the watcher triggers the automatic MT calculation process for the specific event's parameters and settings in an individual thread and sleeps for the waiting interval in order to check for a new event again. The thread that has been triggered to run the calculation process, works in parallel with the main thread of the watcher. Figure 20, presents a flow chart of the watcher's steps.

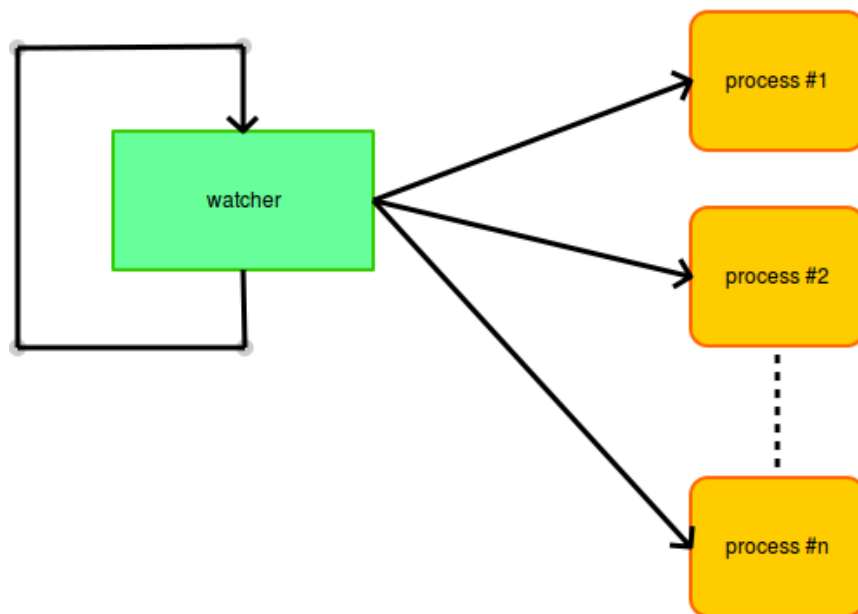


Figure 19. Watcher's process triggering in parallel.

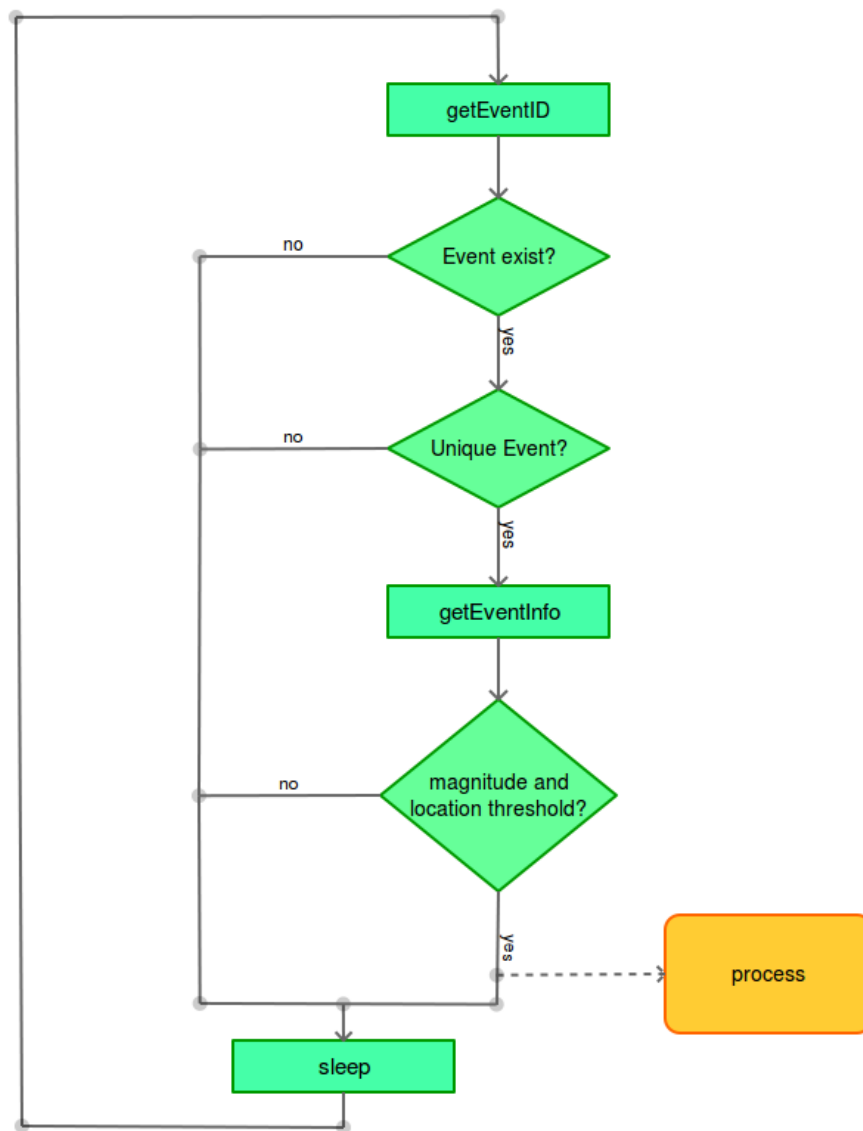


Figure 20. The steps of the watcher's functionality.

4 Case Study

4.1 Evaluation dataset

In order to evaluate the quality of MT solutions produced by scisola, a dataset of 46 earthquakes occurred from 2014-01-02 to 2014-06-25 in Greece was used. The evaluation has been done by comparing the solutions provided by the automatic algorithm of scisola with the respective MT solutions that were calculated by the GI-NOA (Institute of Geodynamics, National Observatory of Athens) in a manual way.

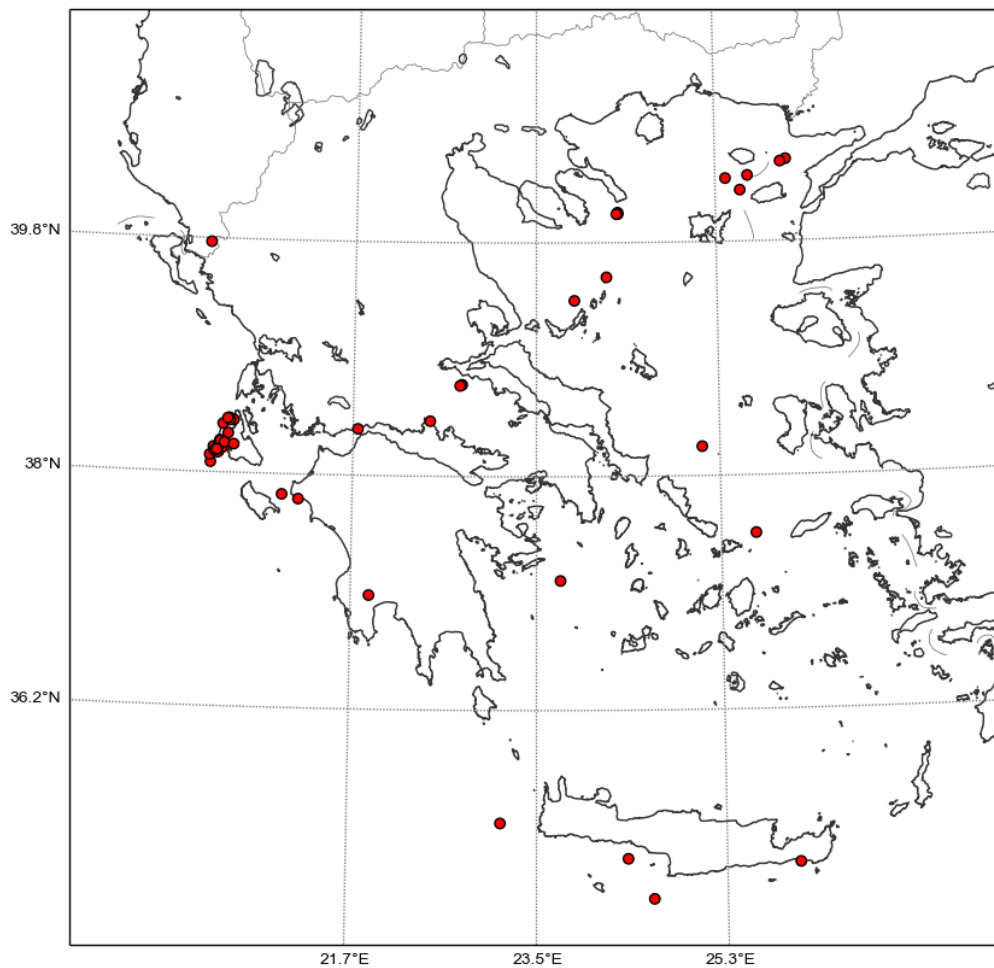


Figure 21. Distribution of evaluation dataset (red dots are epicenters).

For the comparison the Kagan metric has been used. This angle expresses the minimum rotation between two double couple focal mechanisms and is used here as a measure of the automatic solution quality. According to Kagan (Kagan, 1991), minimum angle is 0° (same mechanism) and the maximum value is 120° suggesting maximum divergence between two focal mechanisms. An acceptable agreement is represented by angles of the order of a few tens of degrees, while a strong variance is given by angles larger than 50° - 60° . [6] In figure 21 a map of the geographical distribution of the locations of the analyzed earthquakes is presented. In figure 22 the focal mechanisms of automatic and manual MT solutions by scisola and GINOA respectively are presented. The red focal mechanisms correspond to the automatic MT solutions while the black to the manual ones and the blue colored values represent the Kagan value between the red and black focal mechanism.

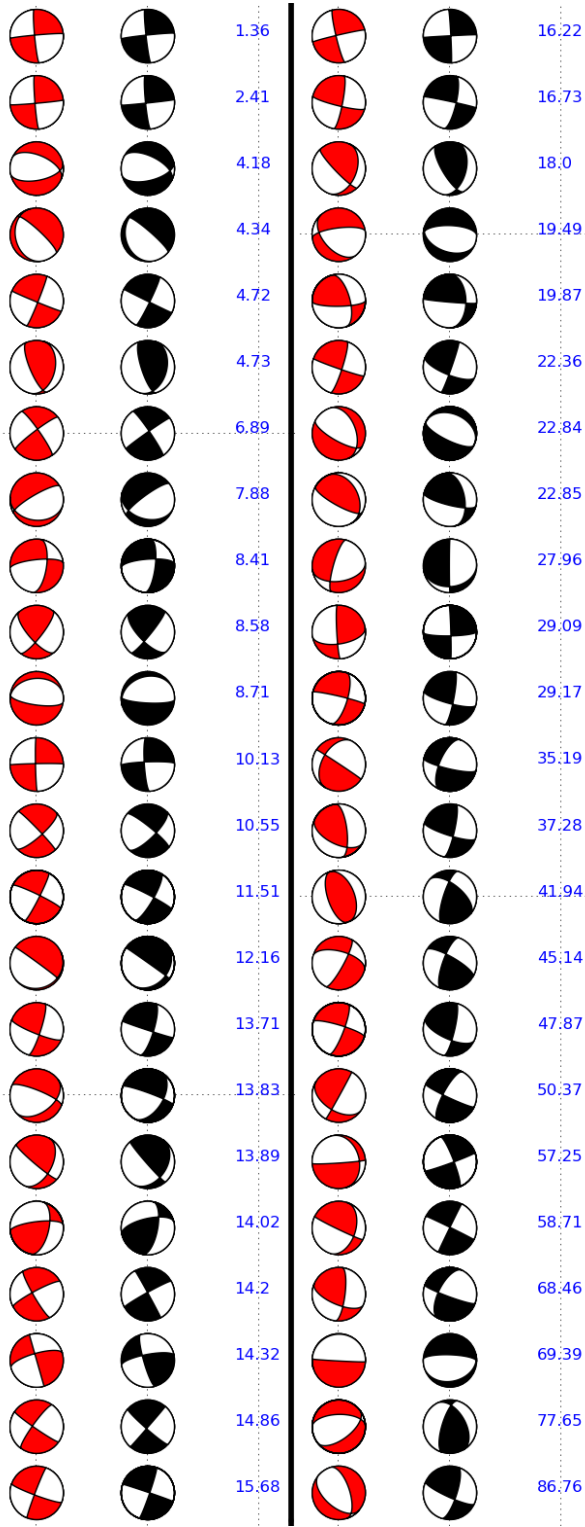


Figure 22. Automatic and manual focal mechanisms of the evaluation dataset.

4.2 *Scisola settings for the evaluation*

The scisola settings were configured as displayed in figure 23.

<i>Scisola Settings</i>	
<i>Variable</i>	<i>Description</i>
center longitude	22.0
center latitude	38.0
distance range (km)	1000
min magnitude	3.5
distance selection	3.5 \leq magnitude \leq 4.0 and 10 \leq distance \leq 100 4.1 \leq magnitude \leq 4.5 and 50 \leq distance \leq 150 4.6 \leq magnitude \leq 5.0 and 80 \leq distance \leq 200 5.1 \leq magnitude \leq 5.5 and 90 \leq distance \leq 250 5.6 \leq magnitude \leq 6.0 and 110 \leq distance \leq 500 6.1 \leq magnitude \leq 12.0 and 330 \leq distance \leq 1000
number of sectors	1
stations per sector	3
edit database button	-
number of sources	20
step search (km)	2
clipping threshold	0.80
crustal model path	(path to the 6 th layer Novotny crustal model)
Start	-75
End	75
step search	2
inversion time	3.5 \leq magnitude \leq 5.5 and tl = 327.68 5.6 \leq magnitude \leq 12.0 and tl = 409.6
inversion frequency	3.5 \leq magnitude \leq 4.2 and frequencies = [0.06, 0.07, 0.09, 0.1] 4.3 \leq magnitude \leq 5.5 and frequencies = [0.04, 0.05, 0.08, 0.09] 5.6 \leq magnitude \leq 6.0 and frequencies = [0.02, 0.03, 0.06, 0.07] 6.1 \leq magnitude \leq 12.0 and frequencies = [0.001, 0.005, 0.01, 0.02]
check interval (sec)	300
process triggering delay (sec)	0
process timeout (sec)	3600

<i>Scisola Settings</i>	
<i>Variable</i>	<i>Description</i>
results folder	-
update database button – reset button	-
seisComP3 path	-
scevtls path	scevtls
scxmldump path	Scxmldump
slinktool path	Slinktool
slinktool host	
slinktool port	-
ISOLA path	-

Figure 23. Scisola’s settings for the evaluation dataset.

4.3 Results

Figure 24, displays a histogram of the Kagan value distribution.

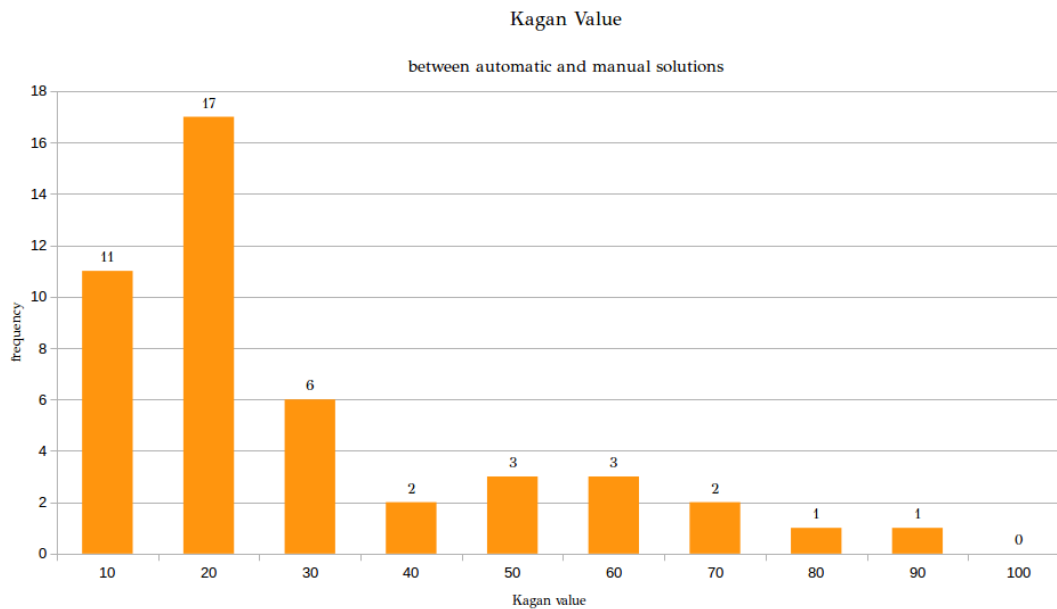


Figure 24. Kagan value histogram.

According to the Kagan angle histogram, the respective automatic and manual MT focal mechanisms of 34 out of 46 events are very similar (Kagan <30). Thus, it produces a success of about 74% on focal mechanism calculation.

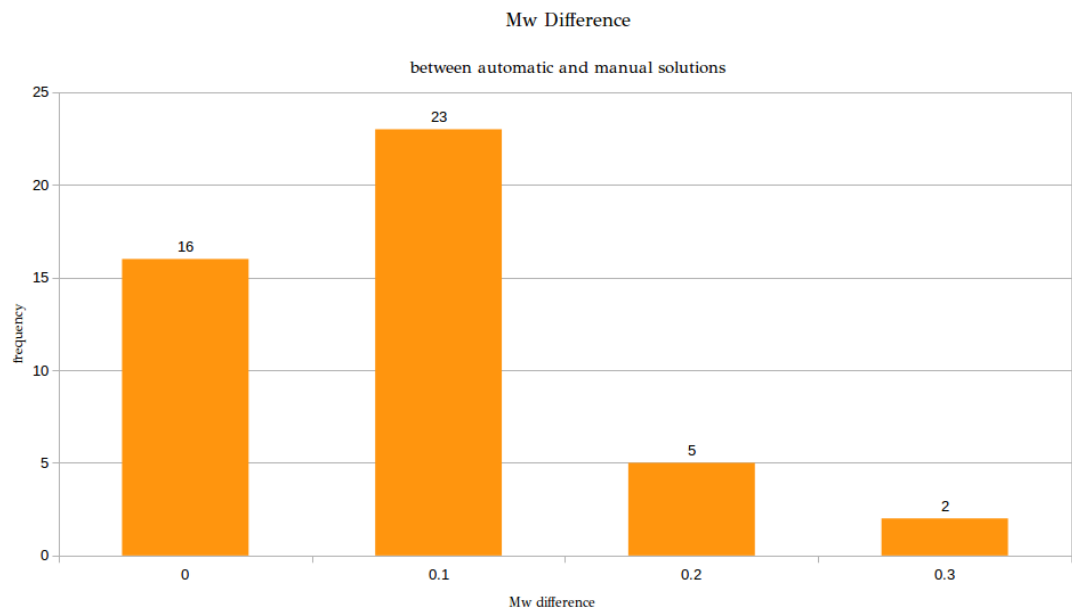


Figure 25. Mw difference histogram.

Figure 25, displays a histogram of the Moment magnitude -Mw- (size of the seismic source) difference between manual and automatic solution. According to this diagram, the respective automatic and manual MT solutions of 39 out of 46 events produce almost the same size of the seismic source (Moment magnitude -Mw-). Thus, it produces a success of about 85% on Mw calculation. Additionally, the maximum Mw difference for the whole evaluation dataset was not more than 0.3 units of Mw.

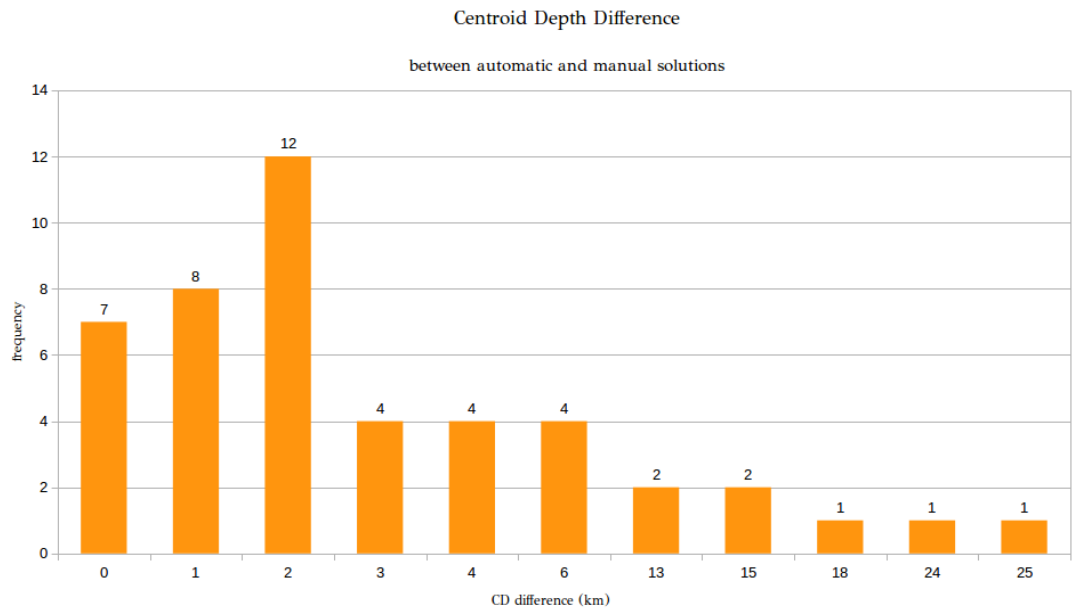


Figure 26. Centroid depth (CD) difference histogram.

The diagram in figure 26, displays a histogram of the Centroid Depth (CD) difference between manual and automatic solution. According to this diagram,

the respective automatic and manual MT solutions of 35 out of 46 events produce almost the same centroid depth of the seismic source. Thus, it produces a success of about 76% on centroid depth calculation.

4.4 *Outcome*

As a very important result, scisola can estimate the size and the depth of the seismic source with adequate accuracy a few minutes after the event, since the overall automatic procedure takes about 5 minutes. This is very crucial for quick estimation of ground motions or tsunami hazard.

It must also be mentioned here that after a quick revision by the user, in just a few minutes, scisola can provide highly accurate solutions.

In general, the error's average value is acceptable, however extreme Kagan values exist and it is important for the automatic procedure evaluation to understand what is causing them.

These higher values are mainly connected:

1. to the edges of the seismic network. Because the contributing to the inversion stations are not distributed very well (large azimuthal gap)
2. to the use of different set of seismic data between automatic and manual
3. to the use of different crustal models between automatic and manual. So far a single crustal model is used in the automatic procedure, the one proposed by (Novotny et al., 2001). While the manual MT solutions are based on region specific crustal models
4. to the presence of noise or disturbances in the records etc

5 User Guide

5.1 Downloading

Scisola is an open-source python based software that can be downloaded through its github webpage (<https://github.com/nikosT/scisola>).

First of all, the user needs to download these 2 folders:

- scisola
- scisola_tools

5.2 Usage

In figure 27, a description of the important files and folders in order to install and run scisola is given.

<i>File/Folder</i>	<i>Description</i>
scisola	contains the scisola source code
scisola/scisola.py	contains the executable file in order to run scisola
scisola_tools/ISOLA	contains the ISOLA source code calibrated for scisola software (actually scisola_tools/ISOLA)
scisola_tools/crustal.dat	a sample of the crustal model that will be used. User can modify it at his own needs
scisola_tools/scisola.sql	the MySQL database schema of scisola database

Figure 27. Scisola's usage files.

5.3 Installation

5.3.1 Python

You need to install these python libraries (some of them are already installed):

- PyQt4 (4.8.4)
- ObsPy (0.8.4)

- matplotlib (1.3.1)
- numpy (1.7.1)
- multiprocessing (0.70a1)
- MySQLdb (1.2.3)
- psycopg2 (2.5.1)
- logging (0.5.1.2)
- distutils (2.7.4)
- decimal (1.7)
- mpl_toolkits
- PIL
- subprocess
- shutil
- codecs
- operator
- os
- time
- sys
- datetime

Python language version: (2.7.4)

5.3.2 ISOLA

You have to compile the ISOLA code by running the compile.sh but you need to have already installed the gfortran at your Linux OS. After running the script, 6 files will be created: gr_xyz, elemse, isola12c, norm12c, dsretc, kagan.

5.3.3 Database

In order to install the database, login from shell to mysql prompt by typing at the linux OS command prompt: mysql -u root -p. After that, on mysql's shell prompt type:

```
create database scisola
```



```
source scisola.sql
```


5.4 *Execution*

If the entire installation process is completed, the user can run scisola software by executing the file scisola/scisola.py by typing in the shell prompt:

```
python scisola.py
```

5.5 *Browsing*

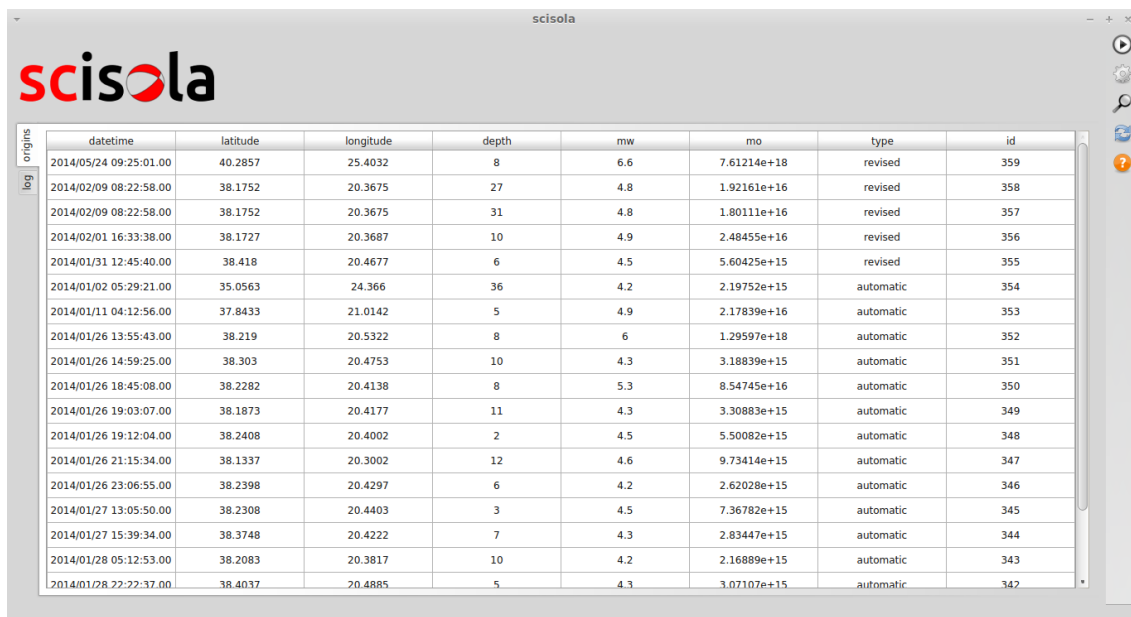
After the successful execution of scisola the user will be prompt with the database log-in screen. The user needs to type the credentials of the scisola and SeisComp3 databases. Usually, the user needs only to provide the password (the same for both tabs). After successful log-in, the user will be prompt with the main window of scisola.



The screenshot shows a window titled "database configuration" with a close button (x) in the top right corner. The window features the "scisola" logo, where "scis" is in red and "ola" is in black. Below the logo, there are two tabs: "scisola" (selected) and "SeisComp3". The "scisola" tab contains several input fields: "type" (a dropdown menu showing "MySQL"), "user" (a text box with "root"), "password" (a text box with "****"), "host" (a text box with "localhost"), "port" (a text box with "3306"), and "database" (a text box with "scisola"). At the bottom left, there is a "status:" label. At the bottom center, there is a green button with a checkmark icon and the text "apply".

Figure 28. Database log-in.

In this window there are two tabs. The origins tab shows the successful MT calculations of the latest 20 events. The log tab shows issues of scisola's functionality, if for example, settings have changed or the watcher has started to listen to SeisComP3 or even if a new incoming event has arrived. At the right panel the user can see five buttons as shown in figure 29. The first button is used for starting and stopping the watcher module. The second button is for scisola configuration. In figure 45 a description and a suggested value for each entry of the settings are shown. The third button is for database search for previous MT calculations, according to defined date-time range as shown in figure 7. By pressing the forth button, the main screen is refreshed with the latest 20 MT calculations. Finally, the fifth button is displaying the version of scisola.



The screenshot shows the scisola application window. On the left, there are two tabs: 'origins' (selected) and 'log'. The 'origins' tab displays a table of 20 MT calculations. The table has columns for datetime, latitude, longitude, depth, mw, mo, type, and id. The 'log' tab is currently empty. On the right side of the window, there is a vertical sidebar with five buttons: a play button (top), a gear icon (configuration), a magnifying glass (search), a refresh button (circular arrow), and a question mark button (version).

datetime	latitude	longitude	depth	mw	mo	type	id
2014/05/24 09:25:01.00	40.2857	25.4032	8	6.6	7.61214e+18	revised	359
2014/02/09 08:22:58.00	38.1752	20.3675	27	4.8	1.92161e+16	revised	358
2014/02/09 08:22:58.00	38.1752	20.3675	31	4.8	1.80111e+16	revised	357
2014/02/01 16:33:38.00	38.1727	20.3687	10	4.9	2.48455e+16	revised	356
2014/01/31 12:45:40.00	38.418	20.4677	6	4.5	5.60425e+15	revised	355
2014/01/02 05:29:21.00	35.0563	24.366	36	4.2	2.19752e+15	automatic	354
2014/01/11 04:12:56.00	37.8433	21.0142	5	4.9	2.17839e+16	automatic	353
2014/01/26 13:55:43.00	38.219	20.5322	8	6	1.29597e+18	automatic	352
2014/01/26 14:59:25.00	38.303	20.4753	10	4.3	3.18839e+15	automatic	351
2014/01/26 18:45:08.00	38.2282	20.4138	8	5.3	8.54745e+16	automatic	350
2014/01/26 19:03:07.00	38.1873	20.4177	11	4.3	3.30883e+15	automatic	349
2014/01/26 19:12:04.00	38.2408	20.4002	2	4.5	5.50082e+15	automatic	348
2014/01/26 21:15:34.00	38.1337	20.3002	12	4.6	9.73414e+15	automatic	347
2014/01/26 23:06:55.00	38.2398	20.4297	6	4.2	2.62028e+15	automatic	346
2014/01/27 13:05:50.00	38.2308	20.4403	3	4.5	7.36782e+15	automatic	345
2014/01/27 15:39:34.00	38.3748	20.4222	7	4.3	2.83447e+15	automatic	344
2014/01/28 05:12:53.00	38.2083	20.3817	10	4.2	2.16889e+15	automatic	343
2014/01/28 22:22:37.00	38.4037	20.4885	5	4.3	3.07107e+15	automatic	342

Figure 29. Main screen.



Figure 30. Main buttons.

By double clicking to a MT calculation, a review window appears such as in figure 32. It consists of 8 tabs. First tab contains solution relative information and a revision panel where the user can change the inversion frequencies or remove a stream by double clicking on it. The second tab, as shown in figure 33, displays the map of the calculated focal mechanism and the contributing stations.

In figure 34, the third tab shows the misfit of observed and synthetic waveforms plot. In figure 35, the forth tab shows the generated plot containing the best focal mechanisms for each depth while in figure 36, the fifth tab shows the generated plot with all the focal mechanisms for each searched depth and for each searched time step. In figure 37, the sixth tab shows the created plot with the contributing, to the inversion, stations. In figure 38, the seventh tab shows a generated text file of the results including the final focal mechanism, which is suitable for distribution over the web. Finally, in figure 39, the eighth tab shows a log file containing the process information throughout the whole computation. At the second, third, fourth and fifth tab, the user can zoom in and out.

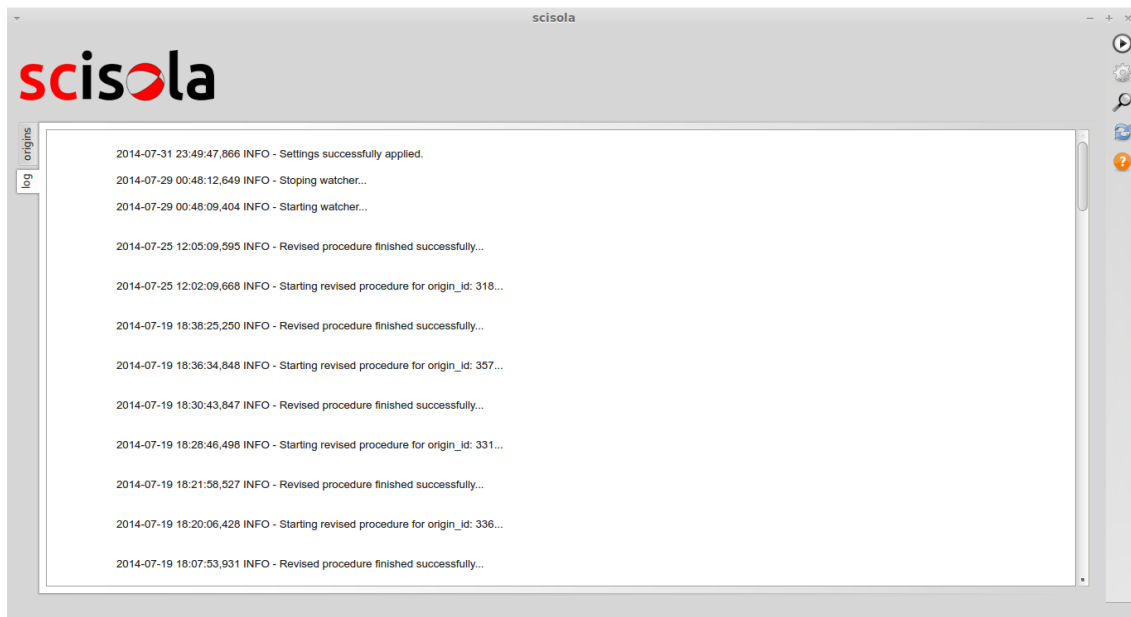


Figure 31. Log screen

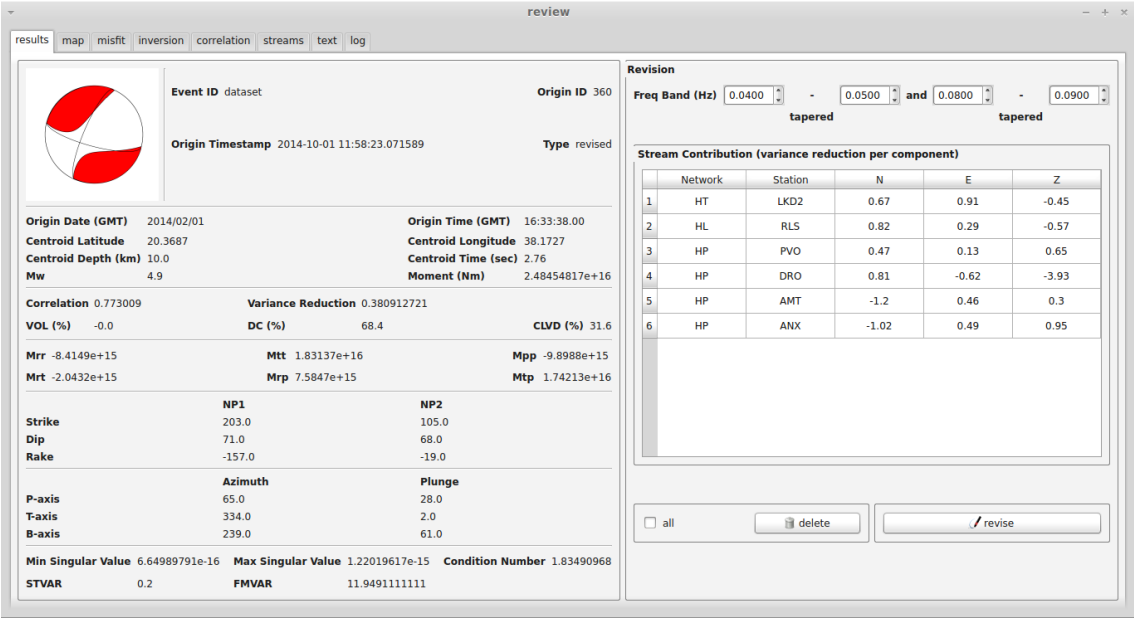


Figure 32. Review screen 1.

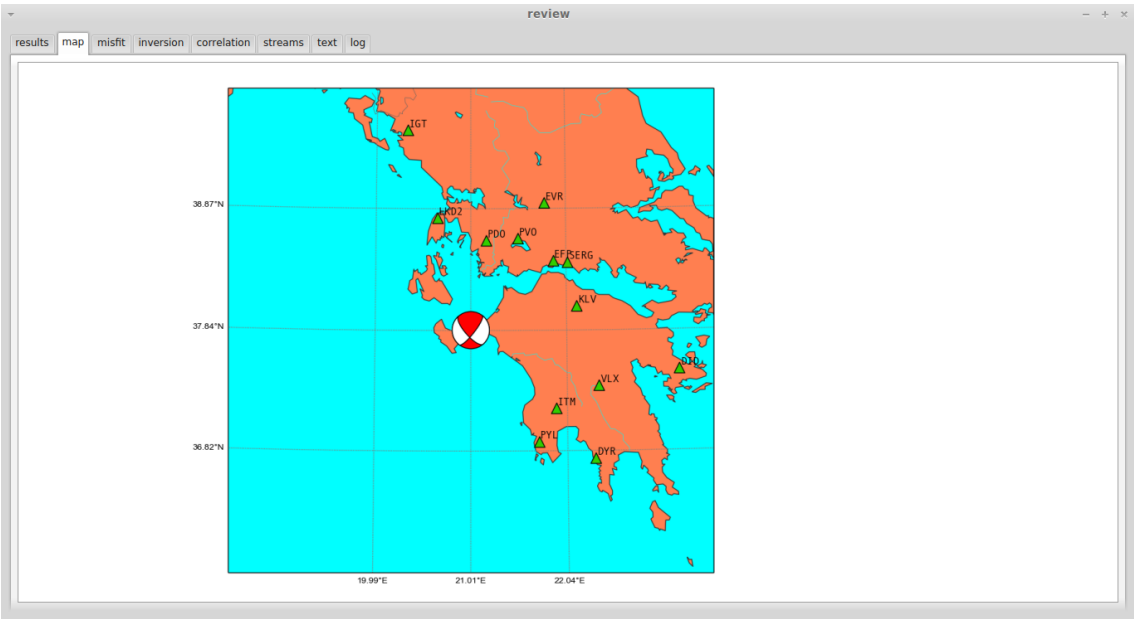


Figure 33. Review screen 2.

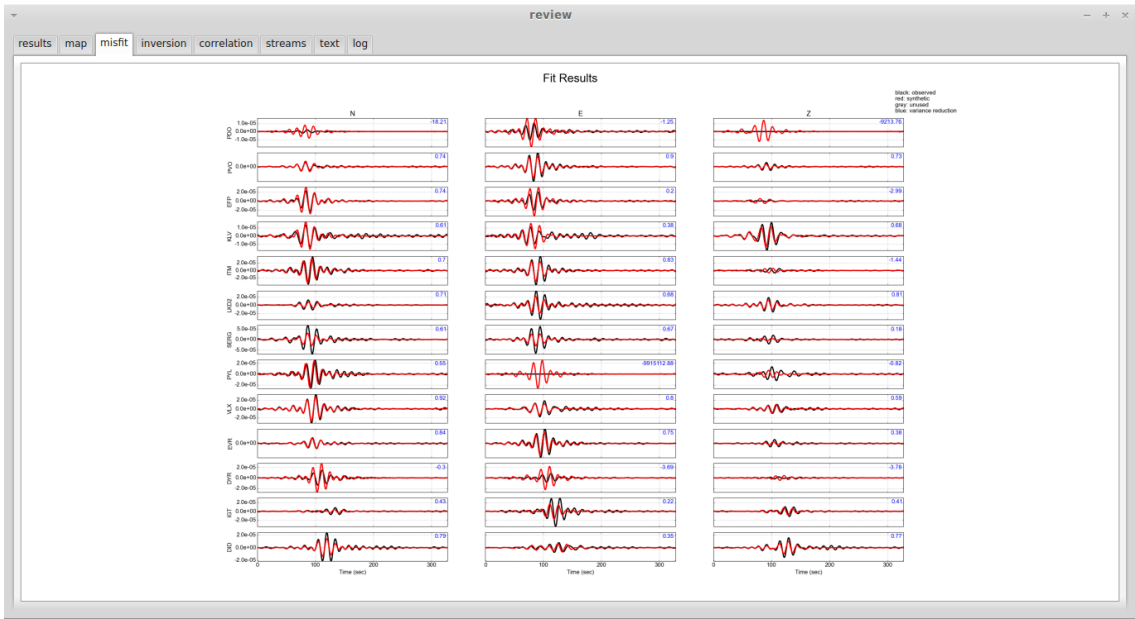


Figure 34. Review screen 3.

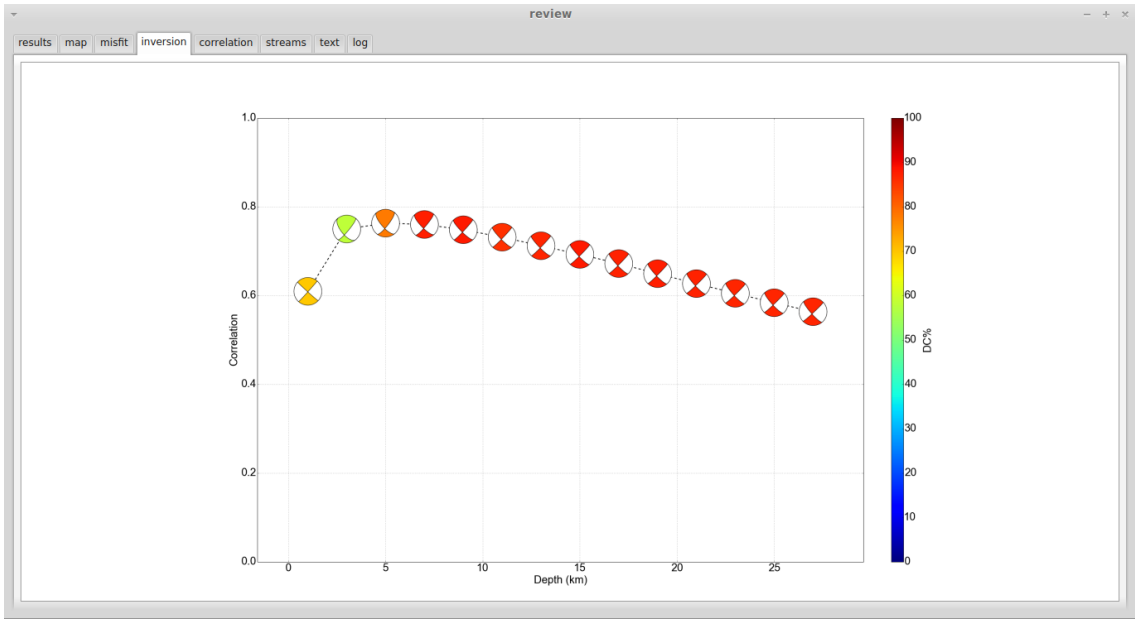


Figure 35. Review screen 4.

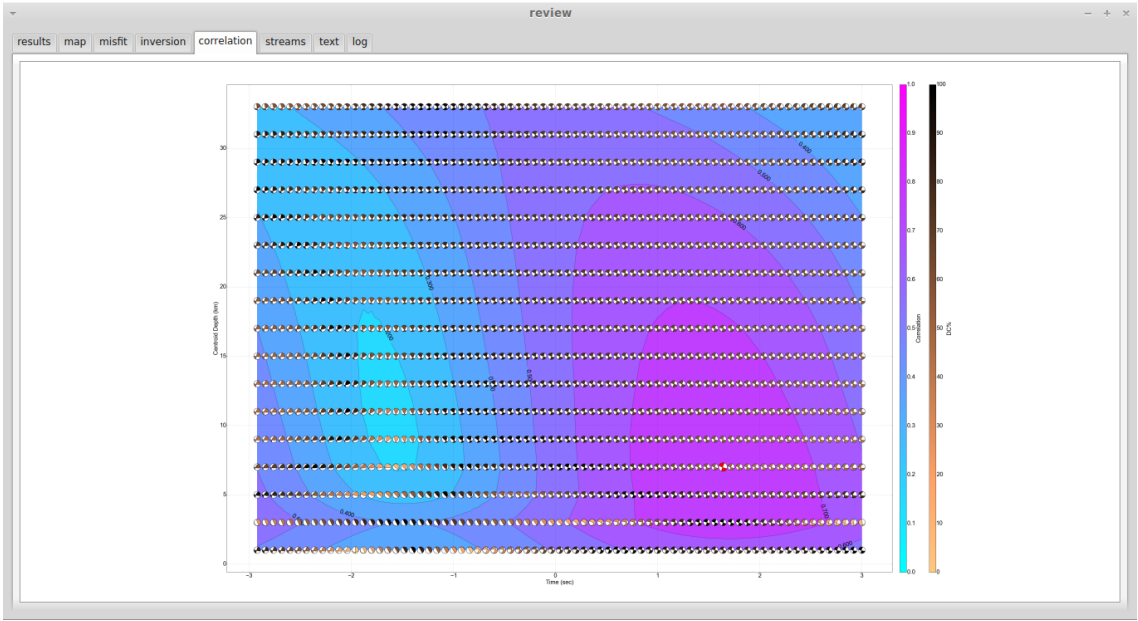


Figure 36. Review screen 5.

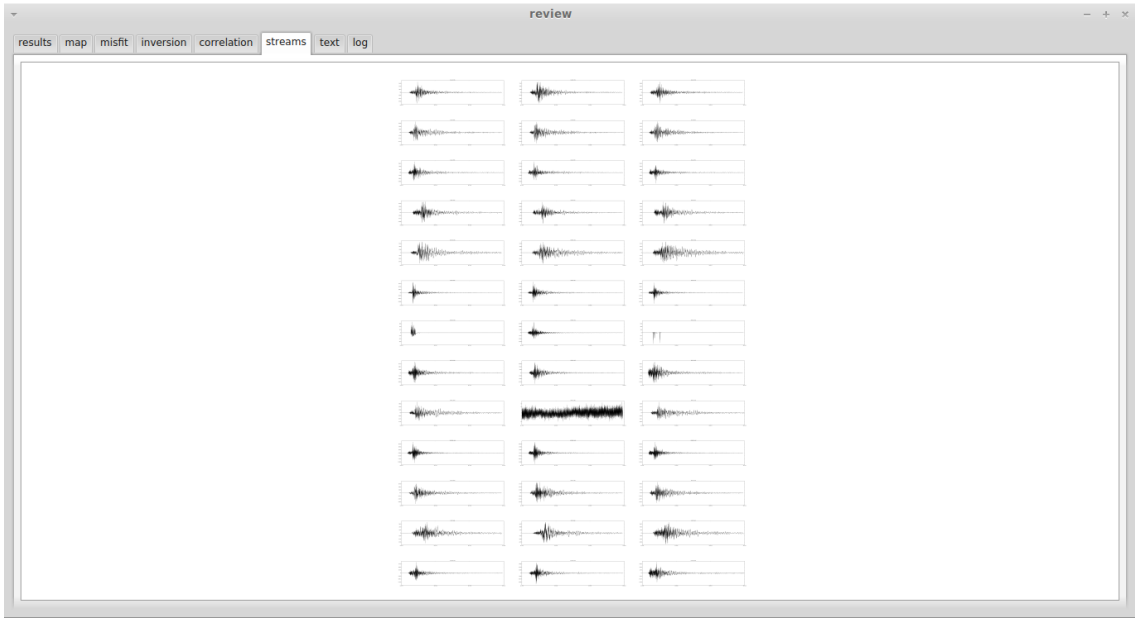


Figure 37. Review screen 6.

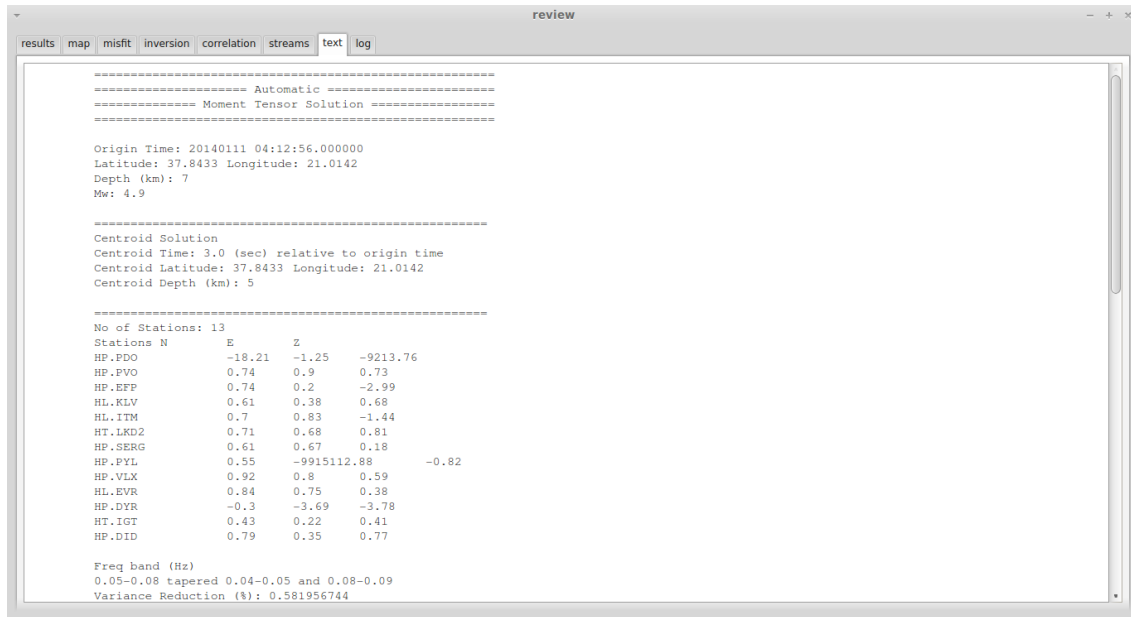


Figure 38. Review screen 7.

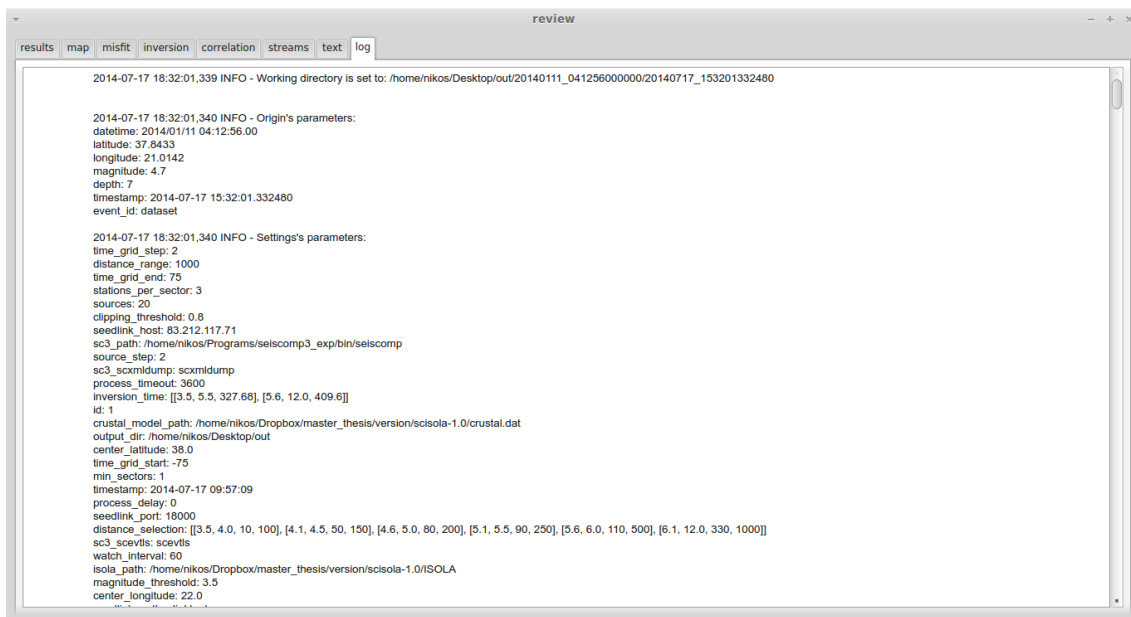


Figure 39. Review screen 8.

5.6 Configuration

In order to run scisola properly, the user needs to configure it. In figure 47, the settings' variables, a description for each variable and their recommended values are presented. For most of the settings the user needs to test scisola manually by running offline datasets of earthquakes to figure out which configuration fits his network. In figures 40, 41, 42 and 43 the four tabs of the scisola's settings screen are presented.

The screenshot shows the 'Settings' window of the Scisola application. It has four tabs: 'stations', 'inversion', 'watcher', and 'settings'. The 'settings' tab is active. The window is divided into several sections:

- Location's restriction:** Contains input fields for 'center longitude' (22.0), 'center latitude' (38.0), and 'distance range (km)' (1000).
- Magnitude's restriction:** Contains a 'min magnitude' dropdown set to 3.5.
- Distance selection:** Contains a table for selecting distance ranges based on magnitude. The table has columns for 'magnitude' (min, max) and 'distance (km)' (min, max). The selected range is 3.5 <= magnitude <= 4.0 and 10 <= distance <= 100. Other ranges include 4.1 <= magnitude <= 4.5 and 50 <= distance <= 150, 4.6 <= magnitude <= 5.0 and 80 <= distance <= 200, 5.1 <= magnitude <= 5.5 and 90 <= distance <= 250, 5.6 <= magnitude <= 6.0 and 110 <= distance <= 500, and 6.1 <= magnitude <= 12.0 and 330 <= distance <= 1000.
- Azimuthal selection:** Contains dropdowns for 'number of sectors' (1) and 'stations per sector' (3).
- Stations:** Contains an 'edit database' button.

At the bottom, there is a status bar showing 'Last changes: 2014-07-31 20:49:46' and an 'apply' button.

Figure 40. Settings screen 1.

stationsinversionwatchersettings

Centroid depth

number of sources20

step search (km)2

Crustal model

crustal model path/home/nikos/Dropbox/master_thesis/version/scisola-1.0/crustal.dat

Inversion time

magnitude

min0.0

max0.0

tl (sec)

16.384

Inversion frequency

magnitude

min0.0

max0.0

frequencies (Hz)

0.00000.00000.00000.0000

Data clipping

clipping threshold0.80

Time grid search

start-75

end75

step search2

3.5 <= magnitude <= 5.5 and tl = 327.68

5.6 <= magnitude <= 12.0 and tl = 409.6

5.6 <= magnitude <= 6.0 and frequencies = [0.02, 0.03, 0.06, 0.07]

6.1 <= magnitude <= 12.0 and frequencies = [0.001, 0.005, 0.01, 0.02]

3.5 <= magnitude <= 4.2 and frequencies = [0.06, 0.07, 0.09, 0.1]

4.3 <= magnitude <= 5.5 and frequencies = [0.04, 0.05, 0.08, 0.09]

Last changes: 2014-07-31 20:49:46

apply

Figure 41. Settings screen 2.

stationsinversionwatchersettings

check interval (sec)

60

process triggering delay (sec)

0

process timeout (sec)

3600

Last changes: 2014-07-31 20:49:46


apply

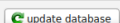
Figure 42. Settings screen 3.

Settings


stations inversion watcher settings


scisola


results folder /home/nikos/Desktop/out 


scisola database  ☐ reset

SeisComP3


SeisComP3 path /home/nikos/Programs/seiscomp3_exp/bin/seiscomp 

scevtls path scevtls 

scxmldump path scxmldump 

slinktool path slinktool  host 83.212.117.71 port 18000

ISOLA

ISOLA path /home/nikos/Dropbox/master_thesis/version/scisola-1.0/ISOLA 


Last changes: 2014-07-31 20:49:46 

Figure 43. Settings screen 4.

stations

	network	station	stream	latitude	longitude	station_priority	stream_priority	azimuth	dip	sensor_gain	datalogger_gain	normalization_factor
531	HP	ITK	HHN	38.0228	22.9673	5	7	0.0	0.0	6000.0	399998.4	571508000.0
532	HP	ITK	HHZ	38.0228	22.9673	5	7	0.0	-90.0	6000.0	399998.4	571508000.0
533	HP	ZKS	HHE	37.696	20.785	5	7	90.0	0.0	1201.0	399998.4	1703690000.0
534	HP	ZKS	HHN	37.696	20.785	5	7	0.0	0.0	1201.0	399998.4	1703690000.0
535	HP	ZKS	HHZ	37.696	20.785	5	7	0.0	-90.0	1201.0	399998.4	1703690000.0
536	HP	SGD	HHE	39.612	20.234	5	7	90.0	0.0	2000.0	399998.4	98533.4
537	HP	SGD	HHN	39.612	20.234	5	7	0.0	0.0	2000.0	399998.4	98533.4
538	HP	SGD	HHZ	39.612	20.234	5	7	0.0	-90.0	2000.0	399998.4	98533.4
539	HP	AXS	HHE	38.1962	21.3763	5	7	90.0	0.0	2000.0	999996.0	98533.4
540	HP	AXS	HHN	38.1962	21.3763	5	7	0.0	0.0	2000.0	999996.0	98533.4
541	HP	AXS	HHZ	38.1962	21.3763	5	7	0.0	-90.0	2000.0	999996.0	98533.4
542	HP	GUR	HHE	37.9363	22.3423	5	7	90.0	0.0	1500.0	999996.0	1.0
543	HP	GUR	HHN	37.9363	22.3423	5	7	0.0	0.0	1500.0	999996.0	1.0
544	HP	GUR	HHZ	37.9363	22.3423	5	7	0.0	-90.0	1500.0	999996.0	1.0
545	HP	RGA	HHE	39.3212	20.3544	5	7	90.0	0.0	2000.0	303030.3	98533.4
546	HP	RGA	HHN	39.3212	20.3544	5	7	0.0	0.0	2000.0	303030.3	98533.4
547	HP	RGA	HHZ	39.3212	20.3544	5	7	0.0	-90.0	2000.0	303030.3	98533.4
548	HP	DSL	HHE	39.1338	21.0964	5	7	90.0	0.0	2000.0	303030.3	98533.4
549	HP	DSL	HHN	39.1338	21.0964	5	7	0.0	0.0	2000.0	303030.3	98533.4




Figure 44. Stations/streams screen.

<i>Scisola Settings</i>		
<i>Variable</i>	<i>Description</i>	<i>Suggested Value</i>
center longitude	The longitude value of the central location for the given distance range, within which events are going to be triggered	-
center latitude	The latitude value of the central location for the given distance range, within which events are going to be triggered	-
distance range (km)	The distance range (in km) value from central location, within which events are going to be triggered	1000
min magnitude	Events with magnitude greater or equal to the min magnitude value are going to be triggered	3.5
distance selection	The distance selection that will be applied according to the rules set by the user. For example, having the suggested set of values, if an earthquake event of 4.7 magnitude arrives, then scisola will choose stations that are within 80 and 200 km from the epicenter's location. Scisola doesn't take rules' order into account. In order to set rules, the user needs to "calibrate" the rules, by testing datasets of events offline and figure out what's the best configuration for them	3.5 \leq magnitude \leq 4.0 and 10 \leq distance \leq 100 4.1 \leq magnitude \leq 4.5 and 50 \leq distance \leq 150 4.6 \leq magnitude \leq 5.0 and 80 \leq distance \leq 200 5.1 \leq magnitude \leq 5.5 and 90 \leq distance \leq 250 5.6 \leq magnitude \leq 6.0 and 110 \leq distance \leq 500 6.1 \leq magnitude \leq 12.0 and 330 \leq distance \leq 1000
number of sectors	The number of the azimuthal sectors that must contain stations in order to execute the inversion procedure	4

<i>Scisola Settings</i>		
<i>Variable</i>	<i>Description</i>	<i>Suggested Value</i>
stations per sector	The maximum number of stations that will be chosen in the azimuthal distribution phase. If the number of stations are more than this value, scisola will remove those stations who have low priority and are farther from the epicenter's location	3
edit database button	This button triggers the stations window where the user can see the streams that are imported to scisola and edit their information	-
number of sources	number of trial sources distributed above and below the automatic depth estimation	20
step search (km)	The step between two adjacent trial sources in depth	2
clipping threshold	The clipping threshold value according to which scisola will decide if a stream is clipped or not	0.80
crustal model path	The path to the crustal model	-
start	The start point of time grid search, given in dt (i.e. sampling interval of data defined by the choice of inversion time)	-75
end	The end point of time grid search	75
step search	The step point of time grid search	2
inversion time	The amount of data (in sec) that will go to inversion according to the rules set by the user. For example, having the suggested set of values, if an earthquake event of 4.7 magnitude arrives, then scisola will choose a time window length (tl) of 327.68 seconds. Scisola doesn't take rules' order into account. In order to set rules, the user needs to "calibrate" the rules, by testing datasets of events and figure out what's the best configuration for them	3.5 ≤ magnitude ≤ 5.5 and tl = 327.68 5.6 ≤ magnitude ≤ 12.0 and tl = 409.6

<i>Scisola Settings</i>		
<i>Variable</i>	<i>Description</i>	<i>Suggested Value</i>
inversion frequency	The inversion frequency that will be used in the inversion according to the rules set by the user. For example, having the suggested set of values, if an earthquake event of 4.7 magnitude arrives, then scisola will choose frequency band for the inversion procedure e.g. 0.04, 0.05, 0.08, 0.09 Hz. Scisola doesn't take rules' order into account. In order to set rules, the user needs to "calibrate" the rules, by testing datasets of events and figure out what's the best configuration for them	3.5 <= magnitude <= 4.2 and frequencies = [0.06, 0.07, 0.09, 0.1] 4.3 <= magnitude <= 5.5 and frequencies = [0.04, 0.05, 0.08, 0.09] 5.6 <= magnitude <= 6.0 and frequencies = [0.02, 0.03, 0.06, 0.07] 6.1 <= magnitude <= 12.0 and frequencies = [0.001, 0.005, 0.01, 0.02]
check interval (sec)	The interval time in seconds that the watcher will check for new incoming event	300
process triggering delay (sec)	The delay time in seconds that the user can assign before triggering an incoming event. This can be used in order to give time to seedlink to gather data at its ringbuffers	0
process timeout (sec)	The time in seconds that if completed, it will terminate the procedure	3600
results folder	The path to the scisola's result folder. Inside this folder scisola will create sub folders by using the origin's date time as a folder name. While inside these sub folders, it will create sub folders by using the date time that triggered the process. Inside these folders the results of the calculation will be saved	-
update database button – reset button	By clicking this button, scisola will retrieve from SeisComP3's database the station/stream information. If reset is enabled, it will delete older streams' and stations' information from the scisola database	-
seisComP3 path	The path to the SeisComP3 folder	-

<i>Scisola Settings</i>		
<i>Variable</i>	<i>Description</i>	<i>Suggested Value</i>
scevtls path	The path to the scevtls module of SeisComP3 folder	scevtls
scxmldump path	The path to the scxmldump module of SeisComP3 folder	scxmldump
slinktool path	The path to the slinktool module of SeisComP3 folder	slinktool
slinktool host	The host value of the slinktool module of SeisComP3	-
slinktool port	The port value of the slinktool module of SeisComP3	-
ISOLA path	The path to the ISOLA folder	-

Figure 45. Scisola settings.

6 Conclusion

6.1 Summary of the master thesis

This master thesis contributes to both seismology and computer engineering by providing the scisola software which is an open-source python based application. It supports the automatic calculation of moment tensors; the seismic events' notifications, stations' information and the corresponding data waveforms are provided by the SeisComP3 system. The moment tensor is calculated through the ISOLA software in parallel mode for much faster calculations. It also supports result storing in a database for a better data management. In addition, it supports extensive configuration changes based on the needs of each researcher, through a user friendly graphical interface. It supports a graphical overview of the moment tensor calculation and the corresponding data fit while it enables the user to revise the moment tensor in case they wish to alter the automatically suggested results. δεξ διορθώσεις πιο πάνω

6.2 Conclusion

Moment tensors, as mentioned before, are used in a wide range of seismological research fields, such as earthquake statistics, earthquake scaling relationships, stress inversion, shakemap generation, tsunami warnings, ground motion evaluation and more. Until now, the procedure of MT calculation as well as the representation and the distribution of its results are a manual and a time consuming operation for most local or regional networks. Thus, the aim of this master thesis was to implement a software that can automatically provide the MT solution of earthquakes that are getting notified by SeisComP3 in real-time. The creation of the scisola

software constitutes a starting point for immediate implementation in the field of seismology and presents a research interest in the field of computer science and engineering too.

6.3 Future Improvements

There are many future improvements that can be implemented in order to make scisola more reliable, fault tolerant, more functional and easier to use. Specifically, some of the main expansions that can be applied are the following:

- Advanced methods and artificial intelligence techniques for selecting stations and streams for the inversion procedure
- Implementation of multiple crustal models based on epicenter's location
- Implementation of advanced signal processing methods to seismic waveforms, in order to avoid various problems such as disturbances, noise and data transmission problems e.g. recently an open-source python code

for automated detection of fling step disturbances

in seismic records has been developed [28] and can be integrated with scisola

- Search of centroid in 3D grid surrounding hypocenter
- Pre-calculation of the Green's functions in order to achieve faster performance
- Better estimation of solution's quality
- Automatic distribution of the results through the web
- Source code improvement by providing more error safe code
- More functionality application according to the needs of the user

Bibliography

- [1] <http://earthquake.usgs.gov/learn/glossary/?term=moment%20tensor>
- [2] http://earthquake.usgs.gov/learn/glossary/images/moment_tensor.gif
- [3] http://en.wikipedia.org/wiki/Focal_mechanism
- [4] http://en.wikipedia.org/wiki/Focal_mechanism#mediaviewer/File:Focal_mechanism_03.jpg
- [5] <http://en.wikipedia.org/wiki/Earthscope>
- [6] Triantafyllis N, Sokos E, and Ilias A, (2013) “Automatic moment tensor determination for the Hellenic Unified Seismic Network.”, *Bulletin of the Geological Society of Greece*, 47.
- [7] Thesis: “Automatic Calculation Procedure of the Seismic Moment Tensor Solution and Distribution of the Results through the Web” Triantafyllis Nikolaos, *Advisor: Prof. Christos Zaroliagis Co-Advisors: As. Prof. Efthimios Sokos and PhD Cand. Aristidis Ilias*
- [8] Triantafyllis, N., Sokos, E., & Ilias, A. (2014). “Scisola: Automatic Moment Tensor Solution for SeisComP3.” *Second European Conference on Earthquake Engineering and Seismology, Istanbul Aug. 25-29, 2014*
- [9] <http://www.seiscomp3.org/doc/seattle/2012.279/>
- [10] http://www.seiscomp3.org/doc/seattle/2012.279/_images/system.png
- [11] <http://seismo.geology.upatras.gr/isola/>
- [12] Sokos, E. N., & Zahradnik, J. (2008). ISOLA a Fortran code and a Matlab GUI to perform multiple-point source inversion of seismic data. *Computers & Geosciences*, 34(8), 967-977.
- [13] <http://en.wikipedia.org/wiki/MySQL>
- [14] http://en.wikipedia.org/wiki/Python_%28programming_language%29

- [15] Beyreuther, M., Barsch, R., Krischer, L., Megies, T., Behr, Y., & Wassermann, J. (2010). “ObsPy: A Python toolbox for seismology.” *Seismological Research Letters*, 81(3), 530-533.
- [16] <http://matplotlib.org>
- [17] <http://matplotlib.org/users/intro.html>
- [18] http://matplotlib.org/_images/contour3d_demo3.png
- [19] <http://www.numpy.org/>
- [20] <https://docs.python.org/2/library/subprocess.html>
- [21] <http://pymotw.com/2/subprocess/>
- [22] <https://docs.python.org/2/library/multiprocessing.html>
- [23] <http://en.wikipedia.org/wiki/PyQt>
- [24] <http://www.riverbankcomputing.co.uk/static/Docs/PyQt4/pyqt-whitepaper-a4.pdf>
- [25] <http://www.riverbankcomputing.co.uk/software/pyqt/intro>
- [26] <http://mysql-python.sourceforge.net/MySQLdb.html>
- [27] <https://pypi.python.org/pypi/psycpg2>
- [28] Vackar, J., Burjanek, J. & Zahradnik, J. (2014). “Automated detection of disturbances in seismic records; MouseTrap code”

