ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ
ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ (Δ.Π.Μ.Σ.)

"ΕΦΑΡΜΟΣΜΕΝΕΣ ΜΑΘΗΜΑΤΙΚΕΣ ΕΠΙΣΤΗΜΕΣ"

ANALYSIS OF METEOROLOGICAL DATA WITH MACHINE LEARNING TECHNIQUES:

A CASE STUDY OF CYCLOGENESIS OVER THE MEDITERRANEAN

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΒΑΣΙΛΕΙΟΥ Ν. ΚΕΛΑΝΤΩΝΗ

ΤΡΙΜΕΛΗΣ ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ:

Π. ΣΤΕΦΑΝΕΑΣ, Επίκουρος Καθηγητής Ε.Μ.Π.
Ι. ΚΟΛΕΤΣΟΣ, Αναπληρωτής Καθηγητής Ε.Μ.Π.
Π. ΨΑΡΡΑΚΟΣ, Καθηγητής Ε.Μ.Π.

ΑΘΗΝΑ, Μάρτιος 2021

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ
ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ (Δ.Π.Μ.Σ.)

"ΕΦΑΡΜΟΣΜΕΝΕΣ ΜΑΘΗΜΑΤΙΚΕΣ ΕΠΙΣΤΗΜΕΣ"

ANALYSIS OF METEOROLOGICAL DATA WITH MACHINE LEARNING TECHNIQUES:

A CASE STUDY OF CYCLOGENESIS OVER THE MEDITERRANEAN

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΒΑΣΙΛΕΙΟΥ Ν. ΚΕΛΑΝΤΩΝΗ

ΕΠΙΒΛΕΨΗ:
Π. ΣΤΕΦΑΝΕΑΣ
Επίκουρος Καθηγητής Ε.Μ.Π.

ΑΘΗΝΑ, Μάρτιος 2021

Σύνοψη

Η λεκάνη της Μεσογείου είναι μία από τις πιο κυκλογενετικές περιοχές του κόσμου. Έχουν γίνει πολλές προσπάθειες στο παρελθόν ώστε να εντοπιστούν οι κύριες περιοχές κυκλογένεσης. Κάνοντας χρήση τεχνικών μηχανικής μάθησης, τέσσερα «καιρικά σενάρια» ανά μήνα υπολογίζονται με σκοπό να εντοπιστούν αυτές οι περιοχές. Το νοτιότερο άκρο της Μεσογείου καταστέλλει την κυκλωνική δραστηριότητα κατά τη διάρκεια του καλοκαιριού, αφήνοντας έτσι το χειμώνα ως την κύρια ενεργή εποχή. Θα επικεντρώσουμε λοιπόν την προσοχή μας στην περίοδο από Οκτώβριο έως και Μάρτιο, και θα την θεωρήσουμε ως τη χειμερινή περίοδο, καθώς τότε έχουμε την μεγαλύτερη συχνότητα εμφάνισης κυκλογενέσεων στην εξεταζόμενη περιοχή.

Χρησιμοποιήθηκαν οι μηνιαίες μέσες τιμές από τα δεδομένα «reanalysis» του ECMWF (Ευρωπαϊκό Κέντρο Μεσοπρόθεσμων Μετεωρολογικών Προγνώσεων), για την περίοδο 41 ετών από το 1979 έως και το 2019, των παρακάτω μετεωρολογικών παραμέτρων:

- Γεωδυναμικό ύψος στα 500hPa
- Πίεση στη μέση στάθμη θάλασσας
- Θερμοκρασία στα 850hPa
- Επιφανειακές ροές Λανθάνουσας και Αισθητής θερμότητας
- Άνεμοι στα 300hPa και 850hPa
- Δυνητικός στροβιλισμός στα 850hPa και στην ισεντροπική επιφάνεια των 315K
- Δυνητική θερμοκρασία στην επιφάνεια των 2PVU

Τα δεδομένα προέρχονται από το ERA-5, που είναι πρόγραμμα της Ευρωπαϊκής Ένωσης και προσφέρονται σε κανονικό πλέγμα γεωγραφικού πλάτους-μήκους με ανάλυση 0.25^{o}. Τα δεδομένα «reanalysis» συνδυάζουν τα αποτελέσματα των αριθμητικών μοντέλων πρόγνωσης με τις παρατηρήσεις από όλο τον κόσμο, ώστε να έχουμε μια πλήρη και συνεπή σειρά δεδομένων κάνοντας χρήση των νόμων της φυσικής. Τα μηνιαία πεδία των μετεωρολογικών παραμέτρων που μελετάμε αποτελούν δισδιάστατα (2D) γεωχωρικά δεδομένα για την περιοχή ενδιαφέροντος (από 25^{o}N έως $50\ ^{o}$N και από $-15\ ^{o}$E έως $45\ ^{o}$E) και αφορούν όλα την ίδια χρονική περίοδο. Επιλέχθηκαν έχοντας κατά νου τις διάφορες διεργασίες τόσο στην ανώτερη όσο και στην κατώτερη τροπόσφαιρα που μπορεί να οδηγήσουν σε επιφανειακή κυκλογένεση στην Μεσόγειο.

Για κάθε μετεωρολογική παράμετρο υπολογίζουμε τον εποχιακό μέσο όρο. Δηλαδή σε κάθε κελί του πλέγματος και για κάθε μήνα, υπολογίζουμε τη μέση τιμή από όλα τα χρόνια. Λαμβάνουμε έτσι 12 εποχιακούς μέσους όρους, έναν για κάθε μήνα του έτους. Στη συνέχεια αφαιρώντας τον αντίστοιχο εποχιακό μέσο (εποχιακό κύκλο) από κάθε μηνιαίο πεδίο, λαμβάνουμε τα μηνιαία πεδία των ανωμαλιών. Ο εποχικός μέσος όρος και τα πεδία των ανωμαλιών για κάθε μετεωρολογική παράμετρο θα μας χρειαστούν για την περεταίρω ανάλυση των δεδομένων με τις μεθόδους «Empirical Orthogonal Functions (EOF)», «Singular Value Decomposition of coupled fields (SVD)» και των τεχνικών κατηγοριοποίησης (Cluster analysis techniques).

Αναλύουμε την χωρική και χρονική διακύμανση κάθε ενός από τα γεωφυσικά πεδία ξεχωριστά με τη χρήση της μεθόδου EOF (Empirical Orthogonal Functions). Η μέθοδος EOF είναι ανάμεσα σε άλλα και ένα εργαλείο μείωσης του όγκου των δεδομένων, όπου υπολογίζονται τα «EOF maps (EOFs)» και τα «Principle Components (PCs)», ώστε στη συνέχεια να εφαρμοστεί αποδοτικά ένας αλγόριθμος κατηγοριοποίησης. Τα EOFs περιγράφουν την χωρική διακύμανση των πεδίων των ανωμαλιών ενώ τα PCs περιγράφουν την εξέλιξη των EOFs στο χρόνο (χρονική διακύμανση). Όπως τα EOFs είναι γραμμικά ανεξάρτητα στο χώρο, τα PCs είναι γραμμικά ανεξάρτητα στο χρόνο. Επιλέγονται τα EOFs και PCs που αντιστοιχούν στις μεγαλύτερες ιδιοτιμές της μεθόδου, και αθροιστικά περιγράφουν τουλάχιστον το 70% της συνολικής διακύμανσης. Με αυτό τον τρόπο μειώνεται σημαντικά ο όγκος των δεδομένων. Επίσης, με αυτά τα EOFs και PCs που

έχουμε επιλέξει μπορούμε να ανακατασκευάσουμε τα αρχικά δεδομένα. Πετυχαίνουμε με αυτό τον τρόπο τον «καθαρισμό» τους καθώς αφαιρούμε τον θόρυβο από αυτά.

Αναλύουμε την χωρική και χρονική συνδιακύμανση πεδίων ανά δύο με τη χρήση της μεθόδου SVD (Singular Value Decomposition). Επιλέγουμε τις σημαντικότερες ιδιάζουσες τιμές που εξηγούν τουλάχιστον το 80% της συνολικής δια-συνδιακύμασης (cross-covariance). Για αυτές, υπολογίζεται ο συντελεστής συσχέτισης Pearson (*r*) μεταξύ της χρονικής δια-συνδιακύμανσης της μιας και της χρονικής δια-συνδιακύμανσης της άλλης μετεωρολογικής παραμέτρου και αντιστοιχούν στην ίδια ιδιάζουσα τιμή. Υψηλή τιμή του (*r*) σημαίνει ισχυρή γραμμική συσχέτιση μεταξύ των εξεταζόμενων μετεωρολογικών παραμέτρων.

Τα χώρο-χρονικά δεδομένα κατηγοριοποιούνται σε τέσσερεις (*k=4*) «κλάσεις» κάνοντας χρήση μίας μη επιτηρούμενης τεχνικής μηχανικής μάθησης. Γίνεται χρήση της μεθόδου *k-means* και *SOM* (Self Organizing Map). Η ανάλυση κατηγοριοποίησης γίνεται στα μηνιαία πεδία των ανωμαλιών του γεωδυναμικού ύψους στα 500hPa που έχουν προβληθεί πάνω στα ιδιοδιανύσματα της μεθόδου EOF που αντιστοιχούν στις μεγαλύτερες ιδιοτιμές και αθροιστικά περιγράφουν τουλάχιστον το 99% της συνολικής διακύμανσης. Ο αλγόριθμος χρησιμοποιείται ώστε να ταξινομήσει τα δεδομένα σε τέσσερεις «κλάσεις» ανά μήνα. Η κατηγοριοποίηση βασίζεται στα πεδία των ανωμαλιών του γεωδυναμικού στα 500hPa διότι αυτό το ισοβαρικό επίπεδο θεωρείται ως το κατευθυντήριο επίπεδο για τα καιρικά συστήματα συνοπτικής κλίμακας στα μέσα πλάτη.

Ο αριθμός των κλάσεων ανά μήνα που έχει επιλεγεί ως τέσσερα (*k=4*), δεν αποτελεί τον βέλτιστο αριθμό. Κάτι τέτοιο ίσως και να μην υπάρχει για τέτοιου είδους σύνθετα, χαοτικά, χώρο-χρονικά δεδομένα. Σκοπός της ανάλυσης σε αυτή την εργασία είναι να επιλεγεί ένας αριθμός κλάσεων αρκετά μεγάλος ώστε να υπάρχει ένας λογικός διαχωρισμός μεταξύ των κέντρων κάθε «κλάσης».

Ο δείκτης που αναφέρεται στην «κλάση» όπου ανήκει ο κάθε μήνας, χρησιμοποιείται για την κατηγοριοποίηση των πεδίων γεωδυναμικού ύψους στα 500hPa και των αντίστοιχων πεδίων των υπολοίπων μετεωρολογικών παραμέτρων. Κάνοντας χρήση αυτών των δεικτών αλλά και των κέντρων κάθε «κλάσης», υπολογίζονται οι αντιπρόσωποι κάθε «κλάσης» για κάθε έναν από τους μετεωρολογικούς παραμέτρους. Σε αυτά τα αντιπροσωπευτικά μοτίβα για κάθε κλάση θα αναφερόμαστε ως «μηνιαία καιρικά σενάρια». Χρησιμοποιώντας σαν μετρική το τετράγωνο της Ευκλείδειας απόστασης εξετάζονται τέσσερεις περιπτώσεις αντιπροσώπων κάθε «κλάσης»:

- Τα πλήρη πεδία που αντιστοιχούν στις παρατηρήσεις γεωδυναμικού στα 500hPa οι οποίες βρίσκονται πλησιέστερα στα κέντρα των κλάσεων.
- Τα πλήρη πεδία, ανακατασκευασμένα μέσω της μεθόδου EOF, που αντιστοιχούν στις παρατηρήσεις γεωδυναμικού στα 500hPa οι οποίες βρίσκονται πλησιέστερα στα κέντρα των κλάσεων.
- Τα πλήρη πεδία που αντιστοιχούν στα κέντρα των κλάσεων του γεωδυναμικού στα 500hPa.
- Αφού γίνει κατηγοριοποίηση των παρατηρήσεων βάση του γεωδυναμικού στα 500hPa, για κάθε μετεωρολογική παράμετρο, βρίσκουμε τα επιμέρους κέντρα. Τα πλήρη πεδία που αντιστοιχούν σε αυτά τα επιμέρους κέντρα είναι τα αντιπροσωπευτικά μοτίβα κάθε κλάσης.

Επιλέγοντας διαφορετικό πλήθος ιδιοδιανυσμάτων που αντιστοιχούν στις N μεγαλύτερες ιδιοτιμές της μεθόδου EOF πάνω στια οποία θα προβάλουμε τα μηνιαία πεδία των ανωμαλιών, μπορεί να επηρεάσει τα αποτελέσματα των αλγορίθμων κατηγοριοποίησης. Θα μελετήσουμε πως η επιλογή του N θα επηρεάσει και στην επιλογή ανάμεσα στις τέσσερεις περιπτώσεις αντιπροσώπων κάθε κλάσης που αναφέραμε παραπάνω, ώστε να περιγράφεται με τον καλύτερο τρόπο το «μέσο μηνιαίο καιρικό σενάριο».

Αυτά τα μηνιαία «καιρικά σενάρια» μας φανερώνουν ότι ο κόλπος της Γένοβας, η Ιβηρική χερσόνησος, η Νότια Ιταλία, η Τυρρηνική θάλασσα, Νότια της Σαρδηνίας, το Ιόνιο πέλαγος, το Αιγαίο πέλαγος, οι Βαλεαρίδες νήσοι και η περιοχή της Κύπρου είναι όντως οι κύριες περιοχές που επηρεάζονται από επιφανειακές κυκλογενέσεις (όπως έχουν δείξει και προγενέστερες μελέτες). Επίσης, η μέθοδος SVD φανερώνει τον υψηλό βαθμό συσχετισμού του πεδίου γεωδυναμικού στα 500hPa, που είναι το πεδίο που βασίστηκε η μέθοδος κατηγοριοποίησης, με τις περισσότερες από τις εξεταζόμενες μετεωρολογικούς παραμέτρους.

Σαν επόμενο βήμα θα ήταν, η ήδη κατηγοριοποιημένη σειρά δεδομένων να χρησιμοποιηθεί σε μία επιτηρούμενη τεχνική μηχανικής μάθησης όπως είναι η CNN (Convolutional neural network). Η μέθοδος CNN θα μπορέσει να προβλέψει σε ποια κατηγορία θα ανήκει στο μέλλον ένα πεδίο γεωδυναμικού στα 500hPa. Τέτοιου είδους μέθοδοι μηχανικής μάθησης μπορούν εν δυνάμει να αποτελέσουν δυνατά εργαλεία για την κατηγοριοποίηση, την ταυτοποίηση και την πρόγνωση μοτίβων σε κλιματικά και περιβαλλοντικά δεδομένα. Λόγω όμως της πολυπλοκότητας αυτών των δεδομένων, που είναι συχνά χώρο-χρονικά, χαοτικά και μη στάσιμα, οι αλγόριθμοι CNN πρέπει να σχεδιάζονται προσεκτικά για κάθε εφαρμογή ξεχωριστά. Παρόλα αυτά, η μέθοδος CNN απαιτεί μιά αρκετά μεγάλη κατηγοριοποιημένη σειρά δεδομένων για να ξεκινήσει. Στην περίπτωσή μας, τα μηνιαία πεδία δεν επαρκούν σε αριθμό ώστε να «εκπαιδευτεί» μία τέτοια τεχνική μηχανικής μάθησης. Θα είχε νόημα εάν εξετάζαμε ημερήσια μοτίβα και το πλήθος του δείγματος μας μετριόταν σε χιλιάδες. Στην περίοδο των 41 ετών που μελετάμε, αντιστοιχούν μόνο 492 μηνιαία πεδία από κάθε μετεωρολογική παράμετρο. Για τους παραπάνω λόγους, η παρουσίαση και αξιολόγηση μίας τέτοιας μεθόδου είναι έξω από το σκοπό αυτής της εργασίας.

Όλα τα ανωτέρω υλοποιήθηκαν με τη χρήσης του πακέτου MATLAB σε αρχεία δεδομένων τύπου NETCDF. Είναι υπό πολλές έννοιες το ιδανικό εργαλείο για την υλοποίηση μεθόδων ανάλυσης που βασίζονται σε πίνακες. Το MATLAB είναι ένα προϊόν της εταιρίας MathWorks και χρησιμοποιείται ευρέως ανάμεσα σε επιστημονικούς ερευνητές.

Περίληψη

Τέσσερα «καιρικά σενάρια» ανά μήνα υπολογίζονται με σκοπό να εντοπιστούν οι κύριες περιοχές επιφανειακών κυκλογενέσεων στη λεκάνη της Μεσογείου. Χρησιμοποιήθηκαν οι μηνιαίες μέσες τιμές από τα δεδομένα «reanalysis» του ECMWF, για την περίοδο 41 ετών 1979-2019, των παρακάτω μετεωρολογικών παραμέτρων:

- Γεωδυναμικό ύψος στα 500hPa
- Πίεση στη μέση στάθμη θάλασσας
- Θερμοκρασία στα 850hPa
- Επιφανειακές ροές Λανθάνουσας και Αισθητής θερμότητας
- Άνεμοι στα 300hPa και 850hPa
- Δυνητικός στροβιλισμός στα 850hPa και στην ισεντροπική επιφάνεια των 315K
- Δυνητική θερμοκρασία στην επιφάνεια των 2PVU

Αναλύουμε την χωρική και χρονική διακύμανση κάθε ενός από τα γεωφυσικά πεδία ξεχωριστά με τη χρήση της μεθόδου EOF (Empirical Orthogonal Functions), και την συνδιακύμανση πεδίων ανά δύο με τη χρήση της μεθόδου SVD (Singular Value Decomposition). Η μέθοδος EOF είναι επίσης και ένα εργαλείο μείωσης του όγκου των δεδομένων, ώστε στη συνέχεια να εφαρμοστεί αποδοτικά ένας αλγόριθμος κατηγοριοποίησης. Η μέθοδος SVD που εφαρμόζεται σε ζεύγη πεδίων φανερώνει το βαθμό συσχέτισης μεταξύ των εξεταζόμενων μετεωρολογικών παραμέτρων.

Τα χώρο-χρονικά δεδομένα κατηγοριοποιούνται σε τέσσερεις (*k=4*) «κλάσεις» κάνοντας χρήση μίας μη επιτηρούμενης τεχνικής μηχανικής μάθησης. Γίνεται χρήση της μεθόδου *k-means* και *SOM* (Self Organizing Map). Η ανάλυση κατηγοριοποίησης γίνεται στα μηνιαία πεδία των ανωμαλιών του γεωδυναμικού ύψους στα 500hPa που έχουν προβληθεί πάνω στα ιδιοδιανύσματα της μεθόδου EOF που αντιστοιχούν στις μεγαλύτερες ιδιοτιμές και αθροιστικά περιγράφουν τουλάχιστον το 99% της συνολικής διακύμανσης. Ο αλγόριθμος χρησιμοποιείται ώστε να ταξινομήσει τα δεδομένα σε τέσσερεις «κλάσεις» ανά μήνα. Ο δείκτης που αναφέρεται στην «κλάση» όπου ανήκει ο κάθε μήνας, χρησιμοποιείται σαν «ταμπέλα» των πεδίων γεωδυναμικού ύψους στα 500hPa και των αντίστοιχων πεδίων των υπολοίπων μετεωρολογικών παραμέτρων. Κάνοντας χρήση αυτών των δεικτών αλλά και των κέντρων κάθε «κλάσης», υπολογίζονται οι αντιπρόσωποι κάθε «κλάσης» για κάθε έναν από τους μετεωρολογικούς παραμέτρους (μηνιαία «καιρικά σενάρια»).

Αυτά τα μηνιαία «καιρικά σενάρια» μας φανερώνουν ότι ο κόλπος της Γένοβας, η Ιβηρική χερσόνησος, η Νότια Ιταλία, η Τυρρηνική θάλασσα, Νότια της Σαρδηνίας, το Ιόνιο πέλαγος, το Αιγαίο πέλαγος, οι Βαλεαρίδες νήσοι και η περιοχή της Κύπρου είναι όντως περιοχές που ευνοούν επιφανειακές κυκλογενέσεις (όπως έχουν δείξει και προγενέστερες μελέτες). Επίσης, η μέθοδος SVD φανερώνει τον υψηλό βαθμό συσχετισμού του πεδίου γεωδυναμικού στα 500hPa, που είναι το πεδίο που βασίστηκε η μέθοδος κατηγοριοποίησης, με τις περισσότερες από τις εξεταζόμενες μετεωρολογικούς παραμέτρους.

Σαν επόμενο βήμα θα ήταν, η ήδη κατηγοριοποιημένη σειρά δεδομένων να χρησιμοποιηθεί σε μία επιτηρούμενη τεχνική μηχανικής μάθησης όπως είναι η CNN (Convolutional neural network). Η μέθοδος CNN θα μπορέσει να προβλέψει σε ποια κατηγορία θα ανήκει στο μέλλον ένα πεδίο γεωδυναμικού στα 500hPa. Η παρουσίαση και αξιολόγηση μίας τέτοιας μεθόδου είναι έξω από το σκοπό αυτής της εργασίας.

Όλα τα ανωτέρω υλοποιήθηκαν με τη χρήσης του πακέτου MATLAB.

Abstract

Four “weather scenarios” per month are computed, in order to reveal areas of surface cyclogenesis across the Mediterranean basin. The data used are the ECMWF’s monthly averaged reanalysis, for the 41 year period 1979-2019, of the following meteorological parameters:

- 500hPa Geopotential height (Z500)
- Mean sea level pressure (MSLP)
- 850hPa Temperature (T850)
- Surface sensible heat flux (SSHF) & Surface latent heat flux (SLHF)
- 300hPa Winds (W300) and 850hPa Winds (W850)
- 850hPa Potential vorticity (PV850) & 315K Potential vorticity (PV315)
- 2PVU Potential Temperature (PT2PVU)

We analyzed the spatial and temporal variability of each of the geophysical fields alone using Empirical Orthogonal Functions (EOF), and in pairs using Singular Value Decomposition (SVD). EOF analysis is also a data reduction tool in order to efficiently apply a clustering algorithm. On the other hand, the SVD of coupled fields reveals the correlations between the meteorological parameters.

The spatio-temporal data is clustered into *k=4* classes using an unsupervised machine learning technique. For that, we use k-means and SOM (Self Organizing Map). The clustering analysis is performed on zonal-mean-removed monthly Z500 anomalies projected on the leading EOF modes that retain at least 99% of the variance. The algorithm is used to classify the data into four (4) clusters per month. The computed cluster index for each month is used to label the full Z500 pattern and the corresponding patterns of the rest meteorological parameters. Using the cluster indices and cluster means (centroids), the cluster representatives of each meteorological parameter are calculated (monthly “weather scenarios”).

Those monthly “weather scenarios” reveal that the Gulf of Genoa, the Iberian Peninsula, southern Italy, the Aegean Sea, the Tyrrhenian Sea, south of Sardinia, the Ionian Sea, the Baleares Islands and Cyprus are indeed areas that favor surface cyclogenesis (as previous studies have noted). Also, the SVD of coupled fields analysis reveals that Z500, which is the field that clustering is based, is highly correlated with most of the meteorological parameters used in this thesis.

As a next step, the labeled dataset could be used to train and test a supervised machine learning technique such as CNN (Convolutional neural network). CNNs can be used to predict which cluster index a Z500 pattern will belong to in the future. Presenting and evaluating a supervised machine learning technique is outside the scope of this thesis.

All the above, were implemented with the use of MATLAB.

Table of contents

Summary - Introduction

The Mediterranean Basin (MB) is one of the most cyclogenetic regions in the world. This is in spite of its being south of the global mid-latitude cyclone belt and the fact that its southern end, latitude 30^{o}N, denotes the northern border of the global desert belt. The southern location of the MB suppresses cyclonic activity in the summer, leaving the winter as the main active season. We will focus our interest in the period October-March as the winter season, since it is the period when most MB cyclones affect the region.

Four "weather scenarios" per month, are computed across the Mediterranean Basin. Each scenario is described by the following parameters:

- 500hPa Geopotential height (Z500)
- Mean sea level pressure (MSLP)
- 850hPa Temperature (T850)
- Surface sensible heat flux (SSHF) & Surface latent heat flux (SLHF)
- 300hPa Winds (W300) and 850hPa Winds (W850)
- 850hPa Potential vorticity (PV850) & 315K Potential vorticity (PV315)
- 2PVU Potential Temperature (PT2PVU)

The data used are ECMWF's monthly averaged reanalysis, for the 41 year period 1979-2019 (section 1.1). Reanalysis combines model data with observations from across the world into a globally complete and consistent dataset using the laws of physics. Each monthly field is 2D geospatial data that span the same period of time. The time steps in our case are months (ex. January 1984). The above meteorological parameters, which are briefly described in sections 1.1.1 - 1.1.7, were chosen having in mind the processes that take place both in upper and lower atmospheric levels and can lead to surface cyclogenesis in the Mediterranean Basin.

For each parameter we calculate the seasonal mean. This is the average value at each grid cell for any given month. (12 seasonal means, one for each month, averaged over the 41 years). Also, we obtain the fields of anomalies, by removing the corresponding seasonal mean (seasonal cycle) from any given month of the dataset. The seasonal mean and the fields of anomalies are needed for further analysis involving Empirical Orthogonal Functions (EOF), Singular Value Decomposition of coupled fields (SVD) and clustering techniques.

We will analyze the spatial and temporal variability of each of the geophysical fields alone using Empirical Orthogonal Functions (EOF) (section 1.2.1), and in pairs using Singular Value Decomposition (SVD) (section 1.2.2). EOF analysis is, amongst others, a data reduction tool that calculates the EOF maps (EOFs) and the Principle Components (PCs) in order to efficiently apply a clustering algorithm. In order to produce optimum quality clusters, the EOF PCs are normalized or standardized before performing the cluster analysis (section 1.2.3). The EOFs are spatial patterns of variability (standing oscillations) and the PCs describe the way EOFs evolve in time (time series) (section 2.1). On the other hand, the SVD of coupled fields will reveal the correlations between the meteorological parameters (section 2.2).

The spatio-temporal data is clustered into *k=4* classes using an unsupervised machine learning technique. For that, we use k-means (section 1.2.4) and SOM (Self Organizing Map, section 1.2.5)

technique, in order to reveal monthly patterns for each of the meteorological parameters. Other unsupervised clustering techniques might also be used such as Hierarchical Clustering, k-Medoids etc. The algorithm is used to classify the data into four (4) clusters per month. The clustering analysis is performed on zonal-mean-removed monthly Z500 anomalies projected on the leading N EOF modes that retain at least 99% of the variance. The clustering is based on the 500hPa geopotential fields because that standard isobaric level is typically understood to be the steering level for synoptic-scale mid-latitude weather systems.

It should be noted that the number of clusters k = 4 is not chosen as an optimal number, which might not even exist for these complex, chaotic, spatio-temporal data. Instead, for the purpose of the analysis here, the chosen k should be large enough such that the cluster centers are reasonably distinct.

The computed cluster index for each month is used to label the full Z500 pattern and the corresponding patterns of the rest meteorological parameters (with the same "time-stamp" as the Z500 field). Using the cluster indices and centroids (the cluster centers), the cluster representatives of each meteorological parameter are calculated. We will refer to those as monthly "weather scenarios". We examine 4 cases of cluster representatives (section 1.2.6):

- Closest to Z500 centroids
- Closest to Z500 centroids (EOF)
- Cluster centroids of Z500
- Cluster centroids of each parameter

In section 1.2.7 we introduce some distance metrics in order to choose the cluster representatives that would describe the "mean weather scenario" in the best way. Choosing different numbers of leading EOF modes (N), to project the zonal-mean-removed monthly anomalies, may affect the results of the clustering algorithms. In section 2.3 we study how the number of leading EOF modes effect on choosing the "best" cluster representatives. The cluster analysis results will be presented in section 2.4.

All the above, were implemented with the use of MATLAB. It is in many ways the ideal tool for matrix–based analysis methods. MATLAB is a product of the MathWorks Company and is widely used among researchers in science and engineering. The scripts used in this thesis are presented in Appendix 4: [MATLAB scripts].

1. Methodology

1.1. The Meteorological Data

The data used in this thesis are from ERA5, the fifth generation ECMWF (European Center for Medium-Range Weather Forecasts) reanalysis for the global climate and weather for the past 4 to 7 decades. Currently data is available from 1979 onwards (final release plus timely updates, this page). ERA5 is a program of the European Commission, branch of Copernicus[1] (Climate Change Service - European eyes on earth).

Reanalysis combines model data with observations from across the world into a globally complete and consistent dataset using the laws of physics. This principle, called data assimilation, is based on the method used by numerical weather prediction centers, where every so many hours (12 hours at ECMWF) a previous forecast is combined with newly available observations in an optimal way to produce a new best estimate of the state of the atmosphere, called analysis, from which an updated, improved forecast is issued. Reanalysis works in the same way, but at reduced resolution to allow for the provision of a dataset spanning back several decades. Reanalysis does not have the constraint of issuing timely forecasts, so there is more time to collect observations, and when going further back in time, to allow for the ingestion of improved versions of the original observations, which all benefit the quality of the reanalysis product.

ERA5 provides hourly estimates for a large number of atmospheric, ocean-wave and land-surface quantities. To facilitate many climate applications, monthly-mean averages have been pre-calculated too.

ERA5 is updated daily with a latency of about 5 days (monthly means are available around the 6th of each month). Data has been re-gridded to a regular lat-lon grid of 0.25 degrees for the reanalysis. There are four main sub sets: hourly and monthly products, both on pressure levels (upper air fields) and single levels (atmospheric, ocean-wave and land surface quantities).

Data are available for download in two file formats, GRIB or NETCDF.

The area of interest is the **Mediterranean Sea** (from 25^{o}N to 50 ^{o}N and from -15 ^{o}E to 45 ^{o}E). Working with MATLAB and NETCDF files for various meteorological variables, EOF analysis, SVD of coupled fields analysis and Clustering analysis is performed. More details about the methods will be presenting in the next chapters.

ERA5 reanalysis monthly averaged data on pressure levels and ERA5 reanalysis monthly averaged data on single levels were downloaded either via the CDS web interface either programmatically using the CDS API service. The following meteorological parameters are used in the present thesis:

1.1.1. Geopotential Height on 500hPa pressure level (Z500)

This parameter is a measure of the height of a point in the atmosphere in relation to its potential energy. It is calculated by dividing the geopotential by the Earth's mean gravitational acceleration, g (=9.80665 m/s^2). The geopotential is the gravitational potential energy of a unit mass,

[1] https://cds.climate.copernicus.eu/cdsapp#!/search?type=dataset

at a particular location, relative to mean sea level. Geopotential is also the amount of work that would have to be done, against the force of gravity, to lift a unit mass to that location from mean sea level.

This parameter plays an important role in synoptic meteorology (analysis of weather patterns). Charts of geopotential height plotted at constant pressure levels (e.g., 300, 500 or 850 hPa) can be used to identify weather systems such as cyclones, anticyclones, troughs and ridges. At the surface of the Earth, this parameter shows the variations in geopotential height of the surface, and is often referred to as the orography.

The units of this parameter are geopotential meters. A geopotential meter is approximately 2% shorter than a geometric meter.

The 500hPa geopotential is usually called the Level of non divergence. A level in the atmosphere throughout which, the horizontal velocity divergence is zero. This is why, that standard isobaric level is typically understood to be the steering level for synoptic-scale mid-latitude weather systems.

1.1.2. Mean Sea Level Pressure (MSLP)

Is the atmospheric pressure at Mean Sea Level, measured in hPa. Its dominant features are the smooth, curving patterns of sea level isolines, which show the central elements of our weather systems: highs, lows and cold fronts. It incorporates the effects of atmospheric processes at higher levels.

In the northern hemisphere, the earth's rotation causes air to flow anticlockwise around low pressure systems and clockwise around high pressure systems. Friction over the earth's surface causes the winds to be deflected slightly inwards towards low pressure centers, and slightly outwards from high pressure systems. Wind strength is inversely proportional to the distance between isolines (the closer the lines, the stronger the winds).

1.1.3. Temperature on 850hPa pressure level (T850)

Is the atmospheric Temperature at constant pressure level of 850hPa, measured in degrees Celsius. The patterns of this parameter helps to identify areas of densely packed isotherms (lines of equal temperature) indicating a front.

1.1.4. Surface Latent Heat Flux (SLHF) & Surface Sensible Heat Flux (SSHF)

“SLHF” is the transfer of latent heat (resulting from water phase changes, such as evaporation or condensation) between the Earth's surface and the atmosphere through the effects of turbulent air motion. Evaporation from the Earth's surface represents a transfer of energy from the surface to the atmosphere.

“SSHF” is the transfer of heat between the Earth's surface and the atmosphere through the effects of turbulent air motion (but excluding any heat transfer resulting from condensation or evaporation). The magnitude of the “SSHF” is governed by the difference in temperature between the surface and the overlying atmosphere, wind speed and the surface roughness. For example, cold air overlying a warm surface would produce a sensible heat flux from the land (or ocean) into the atmosphere

These are both single level parameters and are accumulated over a particular time period which depends on the data extracted. The units are joules per square meter (J/m^2). To convert to watts per square meter (W/m^2), the accumulated values should be divided by the accumulation period expressed in seconds. The ECMWF convention for vertical fluxes is positive downwards.

It is well known that depression formation and deepening are strongly associated with energy transfer mechanisms from the surface of the sea to the lower troposphere[2]. Sensible and latent heat fluxes are the main mechanisms that contribute significantly to energy enrichment of the lower tropospheric layers[3]. The energy for a depression comes from the direct transfer of sensible heat from the warm water into the atmosphere and from the transfer of the latent heat, which is another important source of atmospheric energy. Once vapor molecules become separated from the water's surface, they are swept away by the wind. On rising to higher altitudes where the air is cold, the vapor changes into liquid and ice clouds. During this processes, a great amount of heat energy is released into the environment. This heat provides energy for depressions.

1.1.5. Winds on 300hPa & 850hPa pressure levels (W300 & W850)

The speed of horizontal air movement at constant pressure levels of 850hPa and 300hPa, measured in meters per second. To obtain the magnitude and direction of the horizontal wind, one must combine the U and V component of wind.

The U component of wind is the eastward component of the wind. It is the horizontal speed of air moving towards the east, in meters per second. A negative sign thus indicates air movement towards the west.

The V component of wind is the northward component of the wind. It is the horizontal speed of air moving towards the north, in meters per second. A negative sign thus indicates air movement towards the south.

In the present thesis we use *knots* as units of wind speed.

1.1.6. Potential Vorticity on 850hPa pressure level (PV850) and on 315K potential temperature level (PV315)

Potential vorticity at constant pressure level of 850hPa and at isentropic surface of 315K (an isentropic surface that does not intersect the ground, which is usually the case for the 315 K surface), measured in $K \cdot m^2 \cdot s^{-1} \cdot Kg^{-1}$

Potential vorticity is a measure of the capacity for air to rotate in the atmosphere. If we ignore the effects of heating and friction, potential vorticity is conserved following an air parcel. It is used to look for places where large wind storms are likely to originate and develop. Potential vorticity increases strongly above the tropopause and therefore, it can also be used in studies related to the stratosphere and stratosphere-troposphere exchanges.

Large wind storms develop when a column of air in the atmosphere starts to rotate. Potential vorticity is calculated from the wind, temperature and pressure across a column of air in the atmosphere.

[2] Metaxas, 1978
[3] Cayan, 1992a,b; Repapis et al., 1978

1.1.7. Potential Temperature on 2PVU potential vorticity level (PT2VU)

The potential temperature of a parcel of fluid at pressure P is the temperature that the parcel would attain if adiabatically brought to a standard reference pressure P_0, usually 1,000 hPa. The potential temperature is denoted θ and, for a gas well-approximated as ideal, is given by:

$$\theta = T\left(\frac{P_0}{P}\right)^{R/c_p}$$

where T is the current absolute temperature (in K) of the parcel, R is the gas constant of air, and c_p is the specific heat capacity at a constant pressure. $R/c_p = 2/7$ for air (meteorology).

Potential Temperature measured in degrees Celsius, at level at which potential vorticity (PV) equals 2.0 PV units ($1PVU = 10^{-6} \cdot K \cdot m^2 \cdot s^{-1} \cdot Kg^{-1}$). The height of the dynamic tropopause is commonly taken to be that level. In the troposphere PV is ordinarily below this value and relatively uniform, but in the stratosphere is very much higher due to increased stability[4].

[4] Danielsen 1968; Holton et al. 1995

1.2. The methods of Analysis

In this section we will present you with the methods of analysis that we performed on our data.

1.2.1. Empirical Orthogonal Functions (EOF)

EOF is a data reduction tool but also a method for analyzing the spatial and temporal variability of a single geophysical field, also known as Principal Component Analysis (PCA). The method finds the spatial patterns of variability, their time variation, and gives a measure of the "importance" of each pattern. We will refer to the spatial patterns (standing oscillations) as "EOFs" and the time series as "principal components".

For measurements of some variable at locations x_i $(i = 1, \dots p)$ taken at times t_j $(j = 1, \dots n)$ we store them in a matrix F, as *n* maps each being *p* points long (size of F is *n* by *p*).

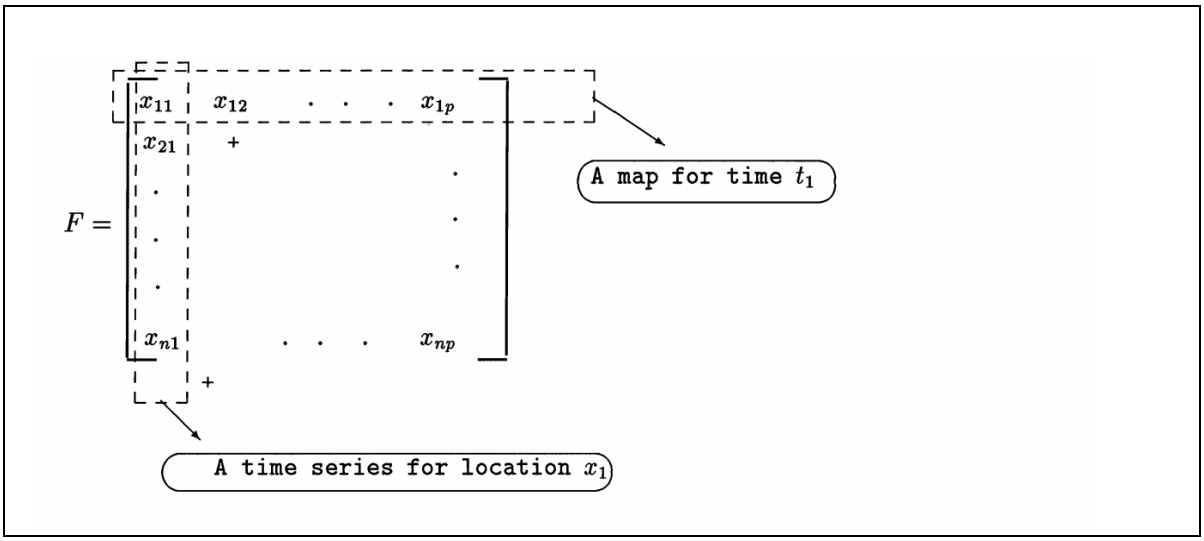


Figure 1: The matrix *F*. Each row is one map, each column is a time series of observations for a given location.

The EOF analysis is performed using F as the data matrix. This way of ordering (time, position) data into a matrix is referred as S-mode analysis.

The data contain quite a bit of seasonal variability that should be removed before EOF analysis because we are not interested in seasonal signals. This is done by removing the seasonal cycle from the data (the matrix *F* is the anomalies matrix). We then remove the mean from each of the *p* time series in F, so that each column has zero mean ("de-trend the data").

We can now form the *covariance matrix* of F by calculating $R = F^T F$, and then we solve the eigenvalue problem:

$$RC = C\Lambda \qquad \textbf{Eq. 1.1}$$

where Λ is a diagonal matrix containing the eigenvalues λ_i of R. The c_i column vectors of C are the eigenvectors of R corresponding to the eigenvalues λ_i. Both Λ and C are of size *p* by *p*.

For each eigenvalue λ_i chosen we find the corresponding eigenvector c_i. Each of these eigenvectors can be regarded as a map. These eigenvectors are the EOFs we are looking for. We arrange the matrices so that the eigenvectors are ordered according to the size of the eigenvalues.

Thus, EOF1, is the eigenvector associated with the biggest eigenvalue and the one associated with the second biggest eigenvalue is EOF2, etc. Each eigenvalue λ_i, gives a measure of the fraction of the total variance in R explained by the mode. This fraction is found by dividing the λ_i by the sum of all the other eigenvalues (the trace of Λ).

The eigenvector matrix C has the property that $C^T C = CC^T = I$ (the identity matrix). This means that EOFs are uncorrelated over space (the eigenvectors are orthogonal to each other, hence the name Empirical Orthogonal Functions).

The pattern obtained when an EOF is plotted as a map, represents a standing oscillation. The time evolution of an EOF shows how this pattern oscillates in time. To see how EOF_j "evolves" in time we calculate:

$$\vec{a}_j = F\vec{c}_j \qquad \textbf{Eq. 1.2}$$

or

$$PC_j = F \cdot EOF_j \qquad \textbf{Eq. 1.3}$$

These are the *principal component time series* (PCs). Just as EOFs where uncorrelated in space, the PCs are uncorrelated in time.

We can reconstruct the data from the EOFs and the PCs as follows:

$$F = \sum_{j=1}^{p} \vec{a}_j \cdot \left(EOF_j\right) \qquad \textbf{Eq. 1.4}$$

A common use of EOFs is to reconstruct a "cleaner" version of the data by truncating this sum at some $j = N \ll p$, using only the first EOFs of the first (largest) few eigenvalues. The rationale is that the first N eigenvectors are capturing the dynamical behavior or the system, and the other eigenvectors (corresponding to the smallest eigenvalues) are just due to random noise.

Usually the EOFs are presented as dimensionless maps, often normalized so that the highest value is 1, or 100. This means that if the associated PCs are also to be presented, then they have to be adjusted correspondingly. The simplest way to do this is to calculate the PCs after having normalized the eigenvector.

Solving a smaller problem (if $n \ll p$)

If the number of observations times (*n*) is much less than the number of grid points (*p*). Since the rank of F is at most *n*, the rank of *R* cannot exceed *n.* So the number of zero eigenvalues of *R* is at least *p-n*. We began with the large eigenvalue problem:

$$RC = C\Lambda \qquad \textbf{Eq. 1.5}$$

Now let $L = FF^T$. The size of L is *n by n,* which in this case is much less than the size of R (*p by p*). Multiplying by F, the right side becomes $FRC = FF^T FC = LFC$. The left hand side becomes $FC\Lambda$.

Now we define $B = FC$ and we rewrite the new smaller eigenvalue problem:

$$LB = B\Lambda \qquad \textbf{Eq. 1.6}$$

The columns of B are not the same as the eigenvectors of R. Λ contains the eigenvalues of L. We solve the eigenvalues for L. Now it remains to find the eigenvectors of R.

Let us assume that we find that the total amount of variance explained by the k largest eigenvalues is large enough so that we can just discard the rest of the eigenvalues. Just as we could find the PCs by taking the projection of F onto the eigenvectors from C, we can find the EOF by taking the projection of F^T on the vectors from B. It turns out that the vectors obtained are proportional to the EOFs of the initial eigenvalue problem. The proportionality factor is $1/\sqrt{\lambda_i}$ (proven easily if we perform Single Value Decomposition on F).

If we are looking for the eigenvector associated with eigenvalue λ_i, we calculate $D = F^T B$. We then find column vector d_i of matrix D, and then our EOF number i is $d_i/\sqrt{\lambda_i}$. This formalism is referred as the "dual formalism", of "sample space" formalism.

<u>In our case</u>

For each variable:

- The seasonal mean is calculated. This is the average value at each grid cell for any given month. (12 seasonal means, one for each month, averaged over the 41 years).
- We obtain the fields of anomalies, by removing the corresponding seasonal mean (seasonal cycle) from any given month of the dataset.
- We "de-trend" the data.
- We form the F matrix
- We perform EOF analysis of the de-trended fields of anomalies.
- We chose the leading N ($N \ll p$) EOF modes whose cumulated variance explain at least 70% of the total variance. For example, for $N = 20$ the cumulated variance of the 20 leading EOF modes of Z500 explain 99.7% of the total variance whereas for the SSHF they explain only 70.3%. We choose N so that the minimum cumulated variance, amongst all meteorological parameters, explains at least 70% of the total variance.
- We scale each PC time series such that it spans the range [-1 1] and adjust correspondingly the associated EOFs.
- We plot the leading EOFs maps (normalized so that the highest value is 1 or 100)
- We study the results

1.2.2. Single Value Decomposition of Coupled Fields (SVD)

The singular Value Decomposition (SVD) method can be thought as a generalization to rectangular matrices of the diagonalization of a square symmetric matrix. It is usually applied in geophysics to two combined data fields, such as Z500 and MSLP. The method identifies pairs of coupled spatial patterns and their temporal variation, with each pair explaining a fraction of the covariance between the two fields. Hence, to perform the SVD method, we construct the temporal cross-covariance matrix between two space and time dependent data fields. This matrix need not be square as the two fields may be defined on a different number of grid points. However, the variables need to span the same period of time. As in EOF method, the temporal means for each variable are removed from the time series at all grid points.

The SVD of the cross-covariance matrix yields two spatially orthogonal sets of singular vectors (spatial patterns analogous to the EOFs) and a set of singular values associated with each pair of vectors (analogous to the eigenvalues). Each pair of spatial patterns describe a fraction of the square covariance (SC) between the two variables. The first pair of patterns describes the largest fraction of the SC and each succeeding pair describes a maximum fraction of the SC that is unexplained by the previous pairs. The square covariance fraction (SCF) accounted for by the *k-th* pair of singular vectors is proportional to the square of the *k-th* singular value. The *k-th* Principle Component (or expansion coefficient) for each variable is computed by projecting the *k-th* singular vector onto the corresponding original data field. The correlation value (*r*) between the *k-th* PCs of the two variables indicates how strongly related the *k-th* coupled patterns are.

How to do it

Let us assume we have two data matrices S and P (the same way we had F in the EOF analysis). We remove the seasonal mean from the data (to remove seasonal variability) and we also remove the mean of each column of S and P so that they are both centered in time. We begin by forming the cross-covariance matrix:

$$C = S^T P \qquad \textbf{Eq. 1.7}$$

Then we perform the singular value decomposition of C. That is, we find the matrices U and V and the diagonal matrix L so that:

$$C = ULV^T \qquad \textbf{Eq. 1.8}$$

The singular vectors of S are the columns of U, and the singular vectors of P are the columns of V. Each pair of singular vectors is a mode of co-variability between the fields S and P. The columns of U are sometimes called left patterns and the columns of V the right patterns. Just as with the EOFs, these patterns represent standing oscillations in the data fields.

We can now find the PCs, which are the time series describing how each mode of variability oscillates in time. For S we calculate:

$$A = SU \qquad \textbf{Eq. 1.9}$$

and for P we calculate:

$$B = PV \qquad \textbf{Eq. 1.10}$$

The columns of the matrices A and B contain the PCs of each mode, and since U and V are both orthogonal, we can reconstruct the data matrices, using $S = AU^T$ and $P = BV^T$. The diagonal of L contains the singular values. The total squared covariance in C is given by the sum of the squared diagonal values of L. This gives a simple way of assessing the relative importance of the singular modes, through the squared covariance fraction (SCF) explained by each mode. If $l_i = L(i,i)$ is the $i-th$ singular value, the fraction of squared covariance (SCF) explained by the corresponding singular vectors $\overrightarrow{u_i}$ and $\overrightarrow{v_i}$ is given by:

$$SCF_i = \frac{{l_i}^2}{\sum {l_i}^2} \qquad \textbf{Eq. 1.11}$$

We calculate the SCF for each singular value and decide how many of these we want to keep. For each mode we deem worthy of attention, we can plot the mode of variability (one for each variable) and the time series of PCs showing how the maps vary in time.

We then calculate the Pearson correlation coefficient (r) for each mode between the PC of the first variable and the PC of the second variable. It has a value between +1 and -1. A value of +1 is total positive linear correlation, 0 is no linear correlation and -1 is total negative linear correlation.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})\,(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \qquad \textbf{Eq. 1.12}$$

Where:

n: is the sample size
$x_i, y_i : are\ the\ individual\ sample\ points\ indexed\ with\ i\ (in\ our\ case, the\ elements\ of\ two\ PCs)$
$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i : the\ sample\ means (in\ our\ case, the\ mean\ of\ each\ PC)$

The correlation coefficient r, tells about the strength and direction of the linear relationship between the two PCs. However the reliability of the linear model also depends on how many observed data points are in the sample.

We perform a hypothesis test of the significance of the correlation coefficient to decide whether the linear relationship in the sample data is strong enough to use to model the relationship in the population. The hypothesis test lets us decide whether the value of the population correlation coefficient ρ is close to zero or significantly different from zero. If the test concludes the correlation coefficient is significantly different from zero, it means that there is significant linear relationship between the two PCs and that we can use the regression line to model the linear relationship between the PCs in the population.

So for the Null hypothesis $\{H_0 : \rho = 0\}$. Meaning that the population correlation coefficient is not significantly different from zero and that there is not a significant linear relationship between the PCs in the population. We calculate the *p-value* with a significance level of 5% ($\alpha = 0.05$).

If the *p-value* is less than the significance level $\alpha = 0.05$, we reject the null hypothesis. There is sufficient evidence to conclude there is significant linear relationship between the PCs because the correlation coefficient is significantly different from zero.

In our case:

For the same de-trended fields of anomalies that we used in the EOF analysis:

- We choose pairs of the calculated fields using all possible combinations.
- We perform SVD of coupled fields analysis.
- We chose the leading modes whose cumulated Square Covariance Fraction explain at least 80% of the total squared covariance.
- We calculate the correlation coefficient (r) and the *p-values.*
- We scale each PC time series such that it spans the range [-1 1]. For plotting and comparing visually the PCs of the variables analyzed.
- We scale each SVD map so that it spans the range [-1 1]. For plotting and comparing visually the SVD maps of the variables analyzed.
- We study the results

1.2.3. Standardization-Normalization of the EOF PCs

In order to produce optimum quality clusters, the EOF PCs are normalized or standardized before performing the cluster analysis.

- We scale each PC time series such that it spans the range [-1 1] :

$$PC_j = \frac{PC_j}{\mathrm{sf}}, j = 1, \dots, N \qquad \textbf{Eq. 1.13}$$

and adjust correspondingly the associated EOFs:

$$EOF_j = EOF_j \cdot \mathrm{sf} \qquad \textbf{Eq. 1.14}$$

scale factor (sf): the maximum value amongst the elements of PC_j

So that $EOF_j \cdot PC_j$ remains the same.

- Also “standard score” (or “z-score”) standardization (standardize using the long-term mean and the standard deviation) was tested.

$$z_j = \frac{PC_j - \mu_j}{\sigma_j}, for\ j = 1, \dots, N \qquad \textbf{Eq. 1.15}$$

μ: $is\ the\ mean\ of\ the\ population$
σ: $is\ the\ standard\ deviation\ of\ the\ population$
N: $number\ of\ EOF\ modes$

In the present thesis, the first case (scaling range [-1 1]) is used for the clustering algorithms. The reason is that when scaling a PC, it is then easier to adjust the associated EOF accordingly.

1.2.4. K-means

The k-means algorithm is an algorithm for partitioning the $\{x^{(n)}\}$ observations into k ($\leq n$) sets:

$$\boldsymbol{S} = \{S_1, S_2, \dots, S_k\} : Sets\ of\ observations$$

Each observation is a p-dimensional real vector (each vector x has p components). Each cluster is parameterized by a vector $m^{(k)}$ called its mean. Each point belongs to the cluster with the nearest mean (cluster centers or cluster Centroids), serving as a prototype of the cluster. K-means clustering minimizes within-cluster variances (squared Euclidean distances), but not regular distances, which would be the more difficult problem. Formally, the objective is to find:

$$\underset{S}{\operatorname{argmin}} \sum_{j=1}^{k} \sum_{x \in S_i} \left\| x - m^{(i)} \right\|^2 = \underset{S}{\operatorname{argmin}} \sum_{j=1}^{k} |S_i|\, Var(S_i)\ , i = 1, \dots, k \qquad \textbf{\textit{Eq. 1.16}}$$

$m^{(i)}$: $is\ the\ mean\ of\ points\ in\ S_i$

To start the k-means algorithm, the means $m^{(k)}$ are initialized in some way, for example to random values (initial seeds). K-means is then an iterative two-step algorithm. In the assignment step, each data point n is assigned to the nearest mean. In the update step, the means are adjusted to match the sample means of the data points that they are responsible for. The K-means algorithm always converges to a fixed point.

In our case:

We wish to find monthly "weather scenarios" that would describe best each month of the year. Clustering is based on the 500hPa Geopotential Height data. We split the 41 year, monthly averaged dataset into 12 sets. Each set corresponds to a month of the year.

$$\boldsymbol{M} = \{M_1, M_2, \dots, M_{12}\}$$

For each of the above sets, we work as follows:

- The number of clusters is taken equal to $k = 4$.
- We form a matrix where the columns are the 500hPa Geopotential Height PCs time series. We chose only the PCs of the leading N ($N \ll p$) EOF modes (explaining at least 99% of the total variance).

  $$A = [PC_1 \quad \cdots \quad PC_N] \in \mathbb{R}^{n \times N} \qquad \textbf{\textit{Eq. 1.17}}$$

  n: $time\ steps\ of\ M_m\ set$, $for\ m = 1, \dots, 12$
  N: $number\ of\ EOF\ modes$
- The lines of that matrix are the observations (N-dimensional real vector).
- We perform k-means algorithm and replicate it 1000 times. Each time with different initial seeds. The clustering with the minimum within-cluster variances is selected.
- The observations of set M_m are now split into k clusters. We use the time steps of the observations of each cluster to label the full 500hPa Geopotential Height patterns with the corresponding cluster number they belong in. The same applies to the rest of the meteorological parameters.

1.2.5. Self Organizing Map Clustering (SOM)

A self-organizing map (SOM) or self-organizing feature map (SOFM) is a type of artificial neural network (ANN) that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map, and is therefore a method to do dimensionality reduction.

SOM learn to classify input vectors according to how they are grouped in the input space. They differ from competitive layers in that neighboring neurons in the SOM learn to recognize neighboring sections of the input space. Thus, self-organizing maps learn both the distribution (as do competitive layers) and topology of the input vectors they are trained on.

Like most artificial neural networks, SOMs operate in two modes: training and mapping. "Training" builds the map using input examples, while "mapping" automatically classifies a new input vector.

The visible part of a self-organizing map is the map space, which consists of components called nodes or neurons. The map space is defined beforehand, usually as a finite two-dimensional region where nodes are arranged in a regular hexagonal or rectangular grid. Each node is associated with a "weight" vector, which is a position in the input space and it has the same dimension as each input vector. While nodes in the map space stay fixed, training consists in moving weight vectors toward the input data (reducing a distance metric) without spoiling the topology induced from the map space. Thus, the self-organizing map describes a mapping from a higher-dimensional input space to a lower-dimensional map space. Once trained, the map can classify a vector from the input space by finding the node with the closest (smallest distance metric) weight vector to the input space vector.

Using SOM algorithm for clustering, we work exactly as we did with the k-means algorithm. The clustering is again based on the 500hPa Geopotential Height data:

- We use 4 neurons (same with the cluster number of the k-means)
- We form a matrix where the columns are the 500hPa Geopotential Height PCs time series. We chose only the PCs of the leading N ($N \ll p$) EOF modes (explaining at least 99% of the total variance).
- The lines of that matrix are the N-dimensional real vector observations.
- We perform SOM algorithm for clustering the above observations into 4 clusters.
- We label the observations as we did with k-means.

1.2.6. The cluster representatives

Either we use k-means or SOM, in order to find the cluster representatives we have to make the following calculations:

- We calculate the “cluster centroids” of 500hPa geopotential height (k for each month). For SOM these are the “weight vectors” of the neurons .These form the matrix C (lines are points in the N-dimensional real space, each line corresponds to a deferent cluster).

$$C = [c_1 \quad \dots \quad c_N] \in \mathbb{R}^{k \times N} \qquad \textbf{\textit{Eq. 1.18}}$$

$$c_j \in \mathbb{R}^k, for\ j = 1, \dots, N$$
$$k: number\ of\ clusters$$
$$N: number\ of\ EOF\ modes$$

- We find the observations closest to the centroids. To do that, we first calculate the squared Euclidian distances of the observations from the cluster centroids:

$$d_i = \sum_{j=1}^{N} \left(PC_j - C_{i,j}\right)^2 \ , for\ i = 1, \dots, k\ and\ j = 1, \dots, N \qquad \textbf{Eq. 1.19}$$

And form the distances matrix:

$$D = [d_1 \quad \dots \quad d_k] \in \mathbb{R}^{n \times k} \qquad \textbf{Eq. 1.20}$$

We then find the minimum of each column of D (in case of multiple minimums in each column, we pick the first occurrence). The observations corresponding to those minimums (with the same time step) are the ones we want.

Then, for each month we examine four (4) cases of cluster representatives. Those representatives will be our monthly "weather scenarios":

a) Closest to Z500 centroids: From the 500hPa Geopotential Height observations closest to the cluster centroids, we find the corresponding time steps. The full patterns at these time steps are used to represent the clusters.
b) Closest to Z500 centroids (EOF): We work as in (a) to find the representatives time steps. For the time steps that we found, we use the EOFs and the corresponding PCs to reconstruct the anomalies fields of each parameter (*Eq. 1.4*). We then add the seasonal mean (as a reverse process of obtaining the anomalies fields from the full pattern) and obtain the full patterns. In this case, choosing only the leading modes, the noise from high frequency phenomena can be removed.
c) Cluster centroids of Z500: The cluster representatives will be the full patterns corresponding to the 500hPa Geopotential Height cluster Centroids.
d) Cluster centroids of each parameter: The cluster representatives for the 500hPa Geopotential Height will be the full patterns corresponding to the cluster Centroids. For each of the rest meteorological parameters, we classify them based on the 500hPa Geopotential Height clustering. We then find their own centroids (means of observations within the 500hPa Geopotential Height clusters). The full patterns corresponding to those centroids are the cluster representatives for each of the rest meteorological parameters.

For the cases (c) and (d), centroids (either centroids of Z500 either centroids of each parameter) are points in the N-dimensional space. We use these points and the corresponding EOFs and PCs to reconstruct the anomalies fields of each parameter (*Eq. 1.4*). We then add the seasonal mean (as a reverse process of obtaining the anomalies fields from the full pattern) and obtain the full patterns.

1.2.7. Choosing the "best" cluster representatives

EOF analysis is a feature extraction method. The more leading EOF modes we choose (N), the cumulated variance explained is closer to the total variance. Also, more features are extracted from the original data, meaning that the clustering algorithm has to cope with that. As a result, for various numbers of EOF modes, the clustering results may vary.

Depending on the EOF modes, firstly we calculate the observations in the N-dimensional space that could potentially serve as cluster representatives. We examine the cases below:

a) Cluster Centroids of Z500: We calculate the "cluster centroids" of 500hPa geopotential height. These form the matrix C (lines are points in the N-dimensional real space, each line corresponds to deferent cluster center).

$$C = [c_1 \quad \dots \quad c_N] \in \mathbb{R}^{k \times N} \qquad \textbf{Eq. 1.21}$$

$c_j \in \mathbb{R}^k, for\ j = 1, \dots, N$
$k: number\ of\ clusters$
$N: number\ of\ EOF\ modes$

b) Closest to Z500 Centroids: We find the 500hPa geopotential height observations time steps that are closest to the centroids calculated in (a). These are the observations that have the minimum squared Euclidean distances (*Eq. 1.19*). At the time steps calculated, for each of the meteorological parameters, we form the matrix $B^{(l)}$ (lines are the observations in the N-dimensional space, each line corresponds to deferent time step).

$$B^{(l)} = [{b^{(l)}}_1 \quad \dots \quad {b^{(l)}}_N] \in \mathbb{R}^{k \times N}, l = 1, \dots, L \qquad \textbf{Eq. 1.22}$$

${b^{(l)}}_j \in \mathbb{R}^k, j = 1, \dots, N$
$k: number\ of\ clusters$
$L: number\ of\ meteorological\ parameters$
$N: number\ of\ EOF\ modes$

c) Cluster centroids of each parameter: For each meteorological parameter, we classify their observations based on the 500hPa Geopotential Height clustering. We then find their own centroids (means of observations within the 500hPa Geopotential Height clusters). These form the matrix $E^{(l)}$ (lines are the points in the N-dimensional space, each line corresponds to deferent cluster). For the case of 500hPa Geopotential Height, it is the same as the C.

$$E^{(l)} = [{e^{(l)}}_1 \quad \dots \quad {e^{(l)}}_N] \in \mathbb{R}^{k \times N}, for\ l = 1, \dots, L \qquad \textbf{Eq. 1.23}$$

${e^{(l)}}_j \in \mathbb{R}^k, for\ j = 1, \dots, N$
$k: number\ of\ clusters$
$L: number\ of\ meteorological\ parameters$
$N: number\ of\ EOF\ modes$

d) "Best Fit" to Parameters Centroids: These are the observations that have the minimum mean distance from the centroids calculated in (c). For each parameter, we calculate the squared Euclidean distances of each observation from the centroids calculated in (c). Then, for each cluster centroid, we sum for all the parameters:

$$\vec{r}_i = \sum_{l=1}^{L}\sum_{j=1}^{N}\left(PC^{(l)}{}_j - C^{(l)}{}_{i,j}\right)^2 \ , i = 1, \dots, k \qquad \textbf{Eq. 1.24}$$

For each of the meteorological parameters, for the time steps that correspond to the minimum of each vector $\vec{r}_i$, we form the matrix $G^{(l)}$ (lines are the observations in the N-dimensional space, each line corresponds to deferent time step).

$$G^{(l)} = \left[g^{(l)}{}_1 \quad \dots \quad g^{(l)}{}_N\right] \in \mathbb{R}^{k\times N} \ , l = 1, \dots, L \qquad \textbf{Eq. 1.25}$$

$g^{(l)}{}_j \in \mathbb{R}^k \ , j = 1, \dots, N$
$k : number\ of\ clusters$
$L : number\ of\ meteorological\ parameters$
$N : number\ of\ EOF\ modes$

Secondly, we have to introduce some metrics in order to choose from the above. For each cluster and each variable, we wish to find an observation that would describe the "mean weather scenario" in the best way. To do that, we calculate the mean Euclidian distance (error) per month:

I. Between the 500hPa geopotential height centroids calculated in (a) and the corresponding centroids of each parameter calculated in (c).

$$ERROR_I(m) = \frac{1}{L}\sum_{l=1}^{L}\left(\frac{1}{k}\sum_{i=1}^{k}\sqrt{\sum_{j=1}^{N}\left(\overrightarrow{c_j} - \overrightarrow{e^{(l)}{}_j}\right)^2}\right) \qquad \textbf{Eq. 1.26}$$

$months:\ m = 1, \dots, 12$

II. Between the observations calculated in (b) and the corresponding centroids of each parameter calculated in (c).

$$ERROR_{II}(m) = \frac{1}{L}\sum_{l=1}^{L}\left(\frac{1}{k}\sum_{i=1}^{k}\sqrt{\sum_{j=1}^{N}\left(\overrightarrow{b^{(l)}{}_j} - \overrightarrow{e^{(l)}{}_j}\right)^2}\right), m = 1, \dots, 12 \qquad \textbf{Eq. 1.27}$$

III. Between the observations calculated in (d) and the corresponding centroids of each parameter calculated in (c).

$$ERROR_{III}(m) = \frac{1}{L}\sum_{l=1}^{L}\left(\frac{1}{k}\sum_{i=1}^{k}\sqrt{\sum_{j=1}^{N}\left(\overrightarrow{g^{(l)}{}_j} - \overrightarrow{e^{(l)}{}_j}\right)^2}\right), m = 1, \dots, 12 \qquad \textbf{Eq. 1.28}$$

IV. Between the 500hPa geopotential height centroids calculated in (a) and the corresponding observations calculated in (b).

$$ERROR_{IV}(m) = \frac{1}{L}\sum_{l=1}^{L}\left(\frac{1}{k}\sum_{i=1}^{k}\sqrt{\sum_{j=1}^{N}\left(\overrightarrow{c_j} - \overrightarrow{b^{(l)}{}_j}\right)^2}\right), m = 1, \dots, 12 \qquad \textbf{Eq. 1.29}$$

2. Results – Discussion

In this section, details of the overall results have been discussed. A complete program using MATLAB has been developed to perform each method of analysis, using the data and theory presented in the previous chapters.

2.1. EOF analysis

An example of application of the EOF analysis method is given in Figure 2. The four leading EOF modes together account for 84.9% of the total monthly Z500 variance. Individually, they explain 41.2%, 23%, 11% and 9.7% of the variance. The spatial patterns associated with these four Z500 modes are shown on the left as dimensionless maps EOF1(Z500), EOF2(Z500), EOF3(Z500) and EOF4(Z500). On the right the temporal variations of each eigenvector, represented by the PC1(Z500), PC2(Z500), PC2(Z500) and PC2(Z500) respectively.

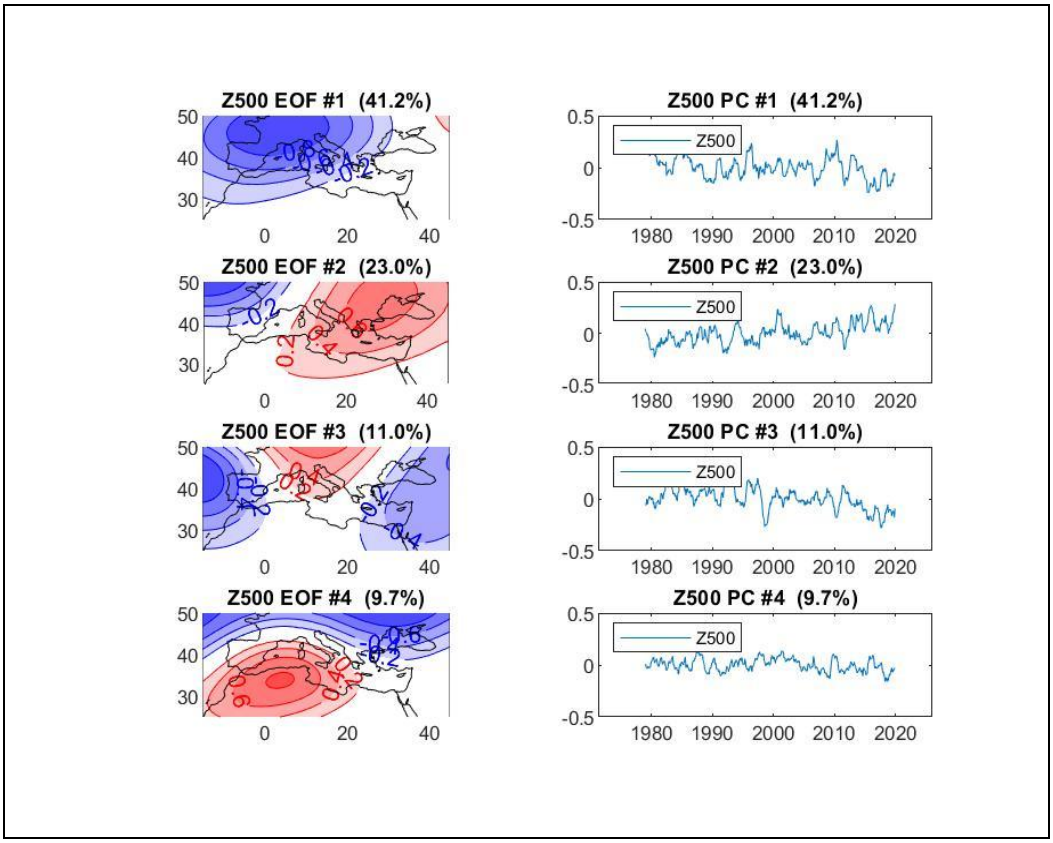


Figure 2: Spatial patterns (EOF maps) and time series (PCs) of the leading four EOF modes for the Z500 fields of anomalies. Spatial patterns are presented as dimensionless maps with amplitudes scaled to range [-1, 1]. Negative contours are blue. Contour interval is 0.2. Time series are smoothed by a 13-month running mean and amplitudes are scaled to range [-1, 1].

EOF1(Z500) exhibits a mono-pole pattern extending in the north west domain. It can be described as the North Atlantic oscillation (500hPa geopotential height increase or decrease in the North West). The Principal Components PC1(Z500) associated with this pattern is dominated by a mixture of inter-annual and decadal fluctuations.

EOF2(Z500) exhibits an oscillating dipole of a Z500 trough in the North East (extending towards the South West) and a Z500 Atlantic ridge. The associated PC time series PC2(Z500) is a mixture of inter-annual and low frequency oscillations.

EOF3(Z500) exhibits three areas of alternating signs. Changes of Z500 in the North are accompanied by changes of the opposite polarity in the East and the West. The associated PC time series PC3(Z500) is dominated mostly by fluctuations with a period of two to three years.

EOF4(Z500) can be described as Z500 changes of the ridge in the South West, accompanied by changes of the opposite polarity in the North East. This oscillates on inter-annual and low frequency timescales, as can be seen from the time series PC3(Z500).

The EOFs and PCs for the rest of the variables are presented in the Appendix 1: [EOF].

2.2. SVD of coupled fields analysis

Table 1 shows the square covariance fraction (SCF) and the correlation coefficients of the leading three SVD modes for the coupled fields of Z500 fields of anomalies, with each of the rest variables (fields of anomalies of PT2PVU, W300, T850, PV850, W850, MSLP, PV315, SSHF, SLHF). The fields are sorted based on the correlation coefficient value, from high to low. All the *p-values* are close to zero (< 0.05) and are not presented in this table.

Table 1: The square covariance fraction (SCF) and the correlation coefficients of the leading three SVD modes for the coupled fields of Z500 fields of anomalies, with each of the rest variables (fields of anomalies of PT2PVU, W300, T850, PV850, W850, MSLP, PV315, SSHF, SLHF).

Z500 Vs …	MODE	SCF %	COR. COEF.
W300	*1*	*55.1*	*0.89*
T850	*1*	*57.3*	*0.89*
MSLP	*1*	*83.2*	*0.88*
PV315	*1*	*58.1*	*0.87*
T850	*2*	*29.4*	*0.86*
T850	*3*	*6.7*	*0.85*
W300	*2*	*27.7*	*0.84*
PV315	*2*	*31.5*	*0.84*
W300	*3*	*8.8*	*0.83*
SSHF	1	55.2	0.83
PT2PVU	1	55.2	0.82
MSLP	2	11.2	0.82
PT2PVU	2	25.2	0.8
SSHF	2	29.2	0.77
PV315	3	4.8	0.76
MSLP	3	3.4	0.75
SLHF	1	64.2	0.75
SLHF	2	29.1	0.74
PT2PVU	3	10	0.73
SSHF	3	9.4	0.73
SLHF	3	3.7	0.58
PV850	1	61.1	0.53
PV850	2	26.6	0.49
PV850	3	8	0.43

We can see that Z500 fields are highly correlated with W300, T850, MSLP and PV315. For example, the coupled Z500-W300 fields show:

- Correlation of 0.89 for the 1st mode, explaining 55.1% of the total square covariance.

- Correlation of 0.84 for the 2nd mode, explaining 27.7% of the total square covariance.
- Correlation of 0.83 for the 3rd mode, explaining 8.8% of the total square covariance.

On the other hand, the strength of the relationship between Z500 and PV850 is the lowest of all. With correlation coefficient of 0.53 for the 1st mode, explaining 61.1% of the total square covariance. The other two modes show even less correlation.

An example of the application of the SVD analysis method is given in Figure 3. Monthly anomalies of Z500 and MSLP are analyzed together over the same spatial and temporal domain. In contrast to the individual EOF analysis performed on the Z500, the SVD analysis on the two combined fields will identify only those modes of behavior in which the Z500 and MSLP variations are strongly coupled.

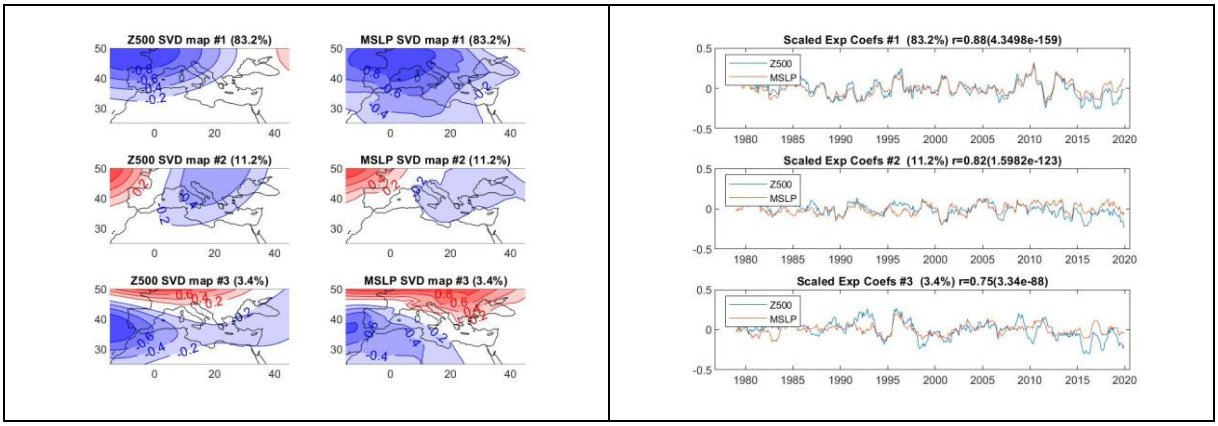


Figure 3: Spatial patterns (SVD maps) and time series (PCs) of the leading three SVD modes for the coupled fields of Z500 and MSLP, fields of anomalies. Spatial patterns are presented as dimensionless maps with amplitudes scaled to range [-1, 1]. Negative contours are blue. Contour interval is 0.2. Time series (Expansion Coefficients or PCs) are smoothed by a 13-month running mean and amplitudes are scaled to range [-1, 1]. Blue lines are the Z500 PCs.

The three leading SVD modes of the coupled Z500 and MSLP variations account for 97.8% of the total square covariance. We label the spatial patterns as SVD map(i) and the time series principle components as PC(i) (the expansion coefficients). We also display the square covariance fractions (SCF) explained by each mode and the correlation coefficient (r) between the PCs of both variables, as indicators of the strength of the coupling. The first mode exhibits a strong and high significant coupling between Z500 and MSLP. By contrast, the coupling coefficient and the SCF associated with the third mode if smaller than those of the other two, which suggests a relatively weaker coupling.

Table 2: Square covariance fraction (SCF) and coupling correlation coefficient between the PCs of Z500 and MSLP, corresponding to the three leading SVD modes.

Mode	Square Cov. Fraction	Correlation
1	83.2%	88%
2	11.2%	82%
3	3.4%	75%

The *p-values* are all close to zero (smaller than the significance level $\alpha = 0.05$). Thus, we reject the null hypothesis. There is sufficient evidence to conclude there is significant linear relationship between the PCs.

We notice that the Z500 patterns associated with the first and second SVD modes repeat the Z500 patterns of the first and second EOF modes (Figure 2), but the second mode is interchanged.

The first coupled mode of variability between Z500 and MSLP can be described as Z500 increase or decrease in the North West, which is accompanied by fluctuations in the same direction of the MSLP in the North West. Both time series are dominated by inter-annual and inter-decadal oscillations.

The second coupled mode of variability can be described as an oscillating dipole of a Z500 trough in the North East (extending towards the South West) and a Z500 Atlantic ridge accompanied by fluctuations in the same direction of the MSLP under the same areas. The time series shows inter-decadal and low frequency oscillations.

The third SVD mode is characterized by fluctuations of the Azores high accompanied by fluctuations of opposite sign of the MSLP over the North domain (extending towards the Balkan Peninsula). The time series shows inter-decadal and low frequency oscillations.

Other examples, involving different pairs of variables, are presented in the Appendix 2: [SVD].

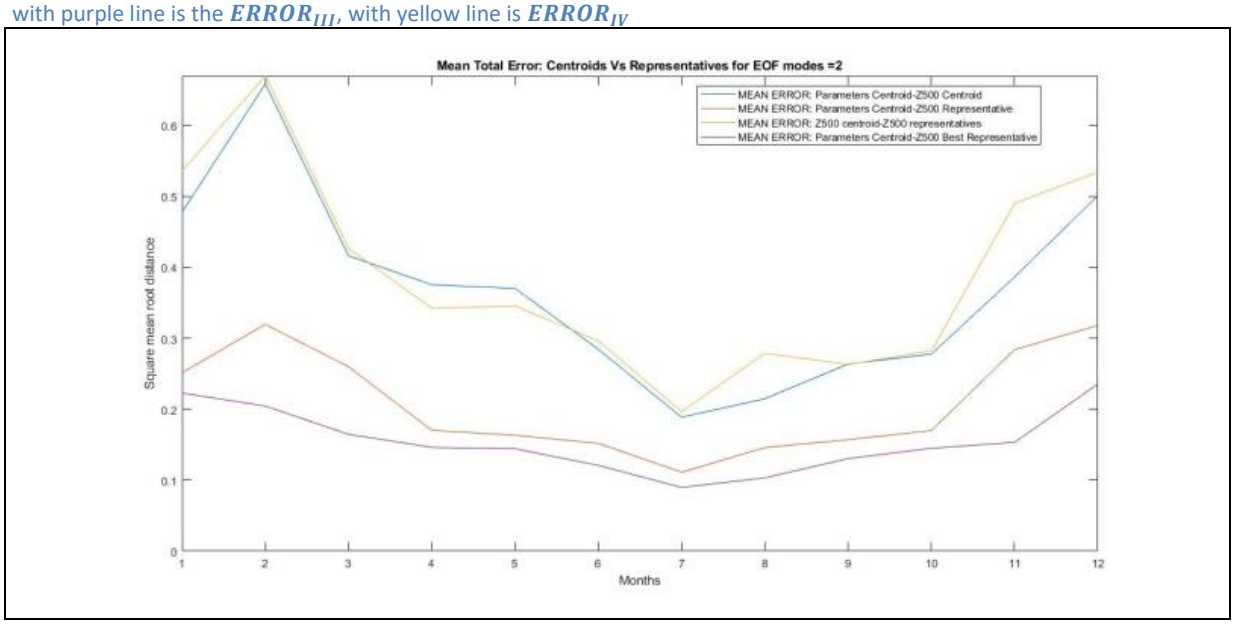
2.3. The cluster representatives analysis

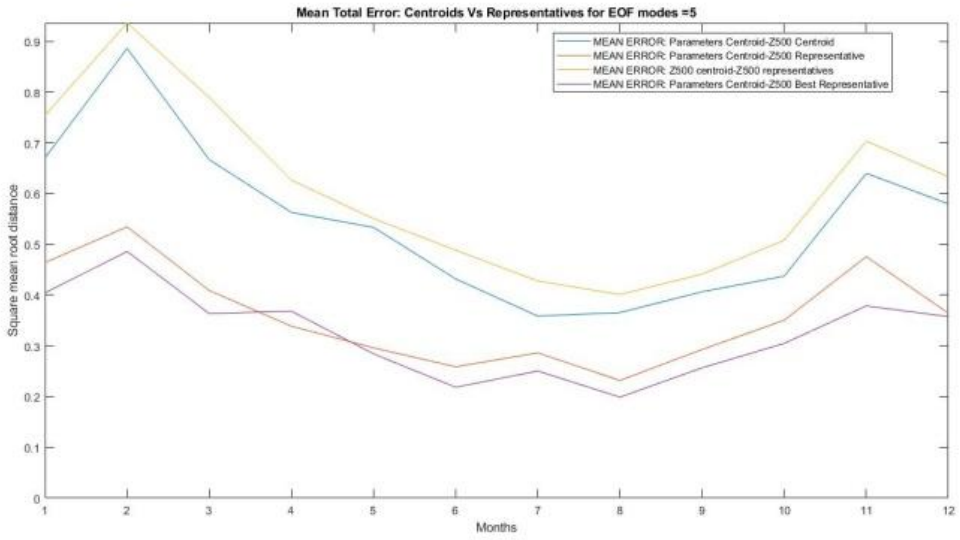
In section 1.2.7 we presented some metrics in order to find the cluster representatives that would describe the "mean weather scenario" in the best way. And by "mean weather scenario" we refer to that represented by the "*Cluster centroids of each parameter*".

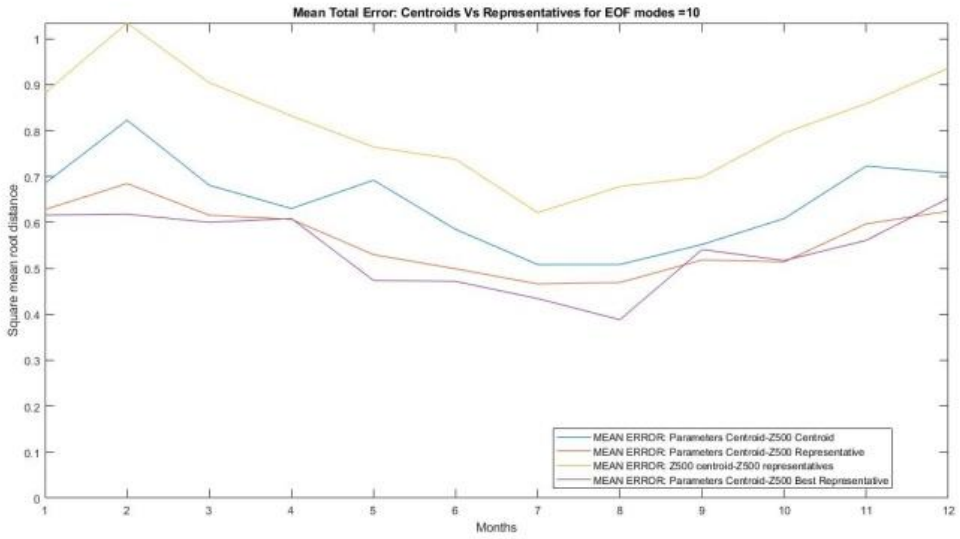
Regardless the choice of EOF modes, the "*Cluster centroids of each parameter*" should be considered as a good choice of cluster representatives. The downside is that the patterns of the meteorological parameters examined, might be inconsistent or out of phase with each other.

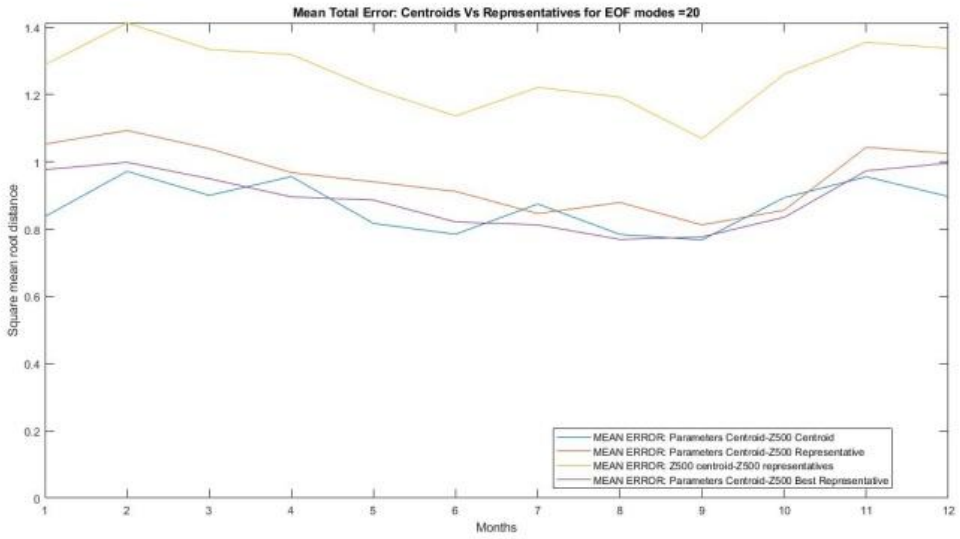
In Table 3, we can see how $ERROR_{I-IV}$ (*Eq. 1.26 - Eq. 1.29*) behaves according to the number of EOF modes we choose.

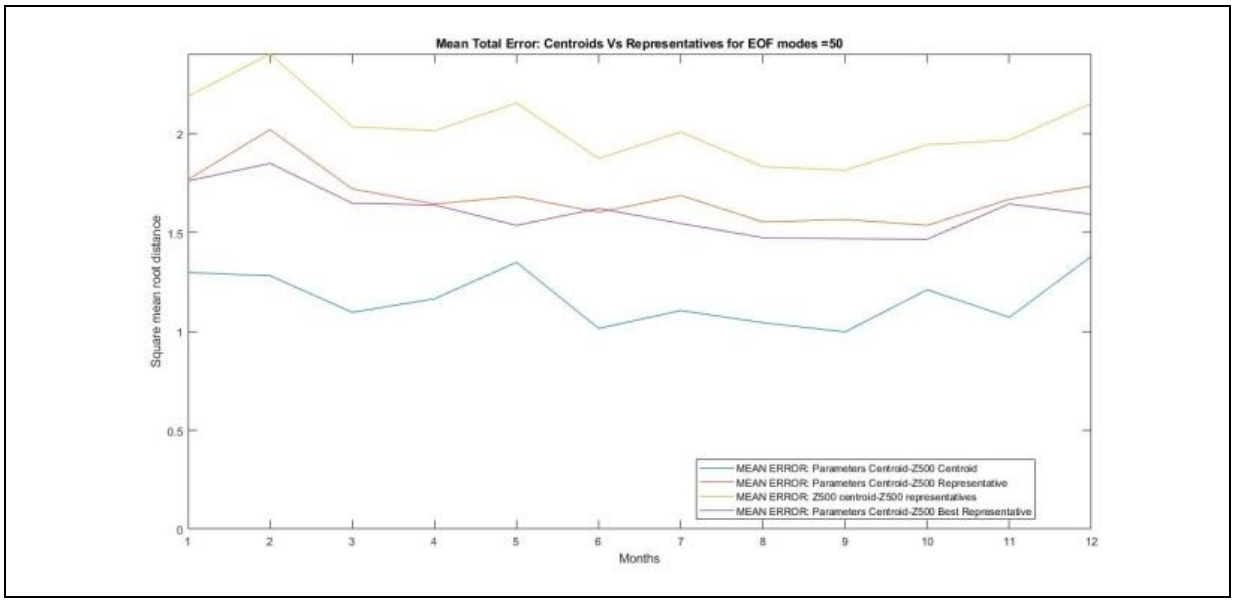
Table 3: For EOF modes = [2, 5, 10, 20, 50], the mean Euclidian errors. With blue line is $\boldsymbol{ERROR_I}$, with red line is $\boldsymbol{ERROR_{II}}$, with purple line is the $\boldsymbol{ERROR_{III}}$, with yellow line is $\boldsymbol{ERROR_{IV}}$




Mean Total Error: Centroids Vs Representatives for EOF modes =5
MEAN ERROR: Parameters Centroid-Z500 Centroid
MEAN ERROR: Parameters Centroid-Z500 Representative
MEAN ERROR: Z500 centroid-Z500 representatives
MEAN ERROR: Parameters Centroid-Z500 Best Representative
Square mean root distance
Months


Mean Total Error: Centroids Vs Representatives for EOF modes =10
MEAN ERROR: Parameters Centroid-Z500 Centroid
MEAN ERROR: Parameters Centroid-Z500 Representative
MEAN ERROR: Z500 centroid-Z500 representatives
MEAN ERROR: Parameters Centroid-Z500 Best Representative
Square mean root distance
Months


Mean Total Error: Centroids Vs Representatives for EOF modes =20
MEAN ERROR: Parameters Centroid-Z500 Centroid
MEAN ERROR: Parameters Centroid-Z500 Representative
MEAN ERROR: Z500 centroid-Z500 representatives
MEAN ERROR: Parameters Centroid-Z500 Best Representative
Square mean root distance
Months



For $N = 50$ (choosing the leading 50 EOF modes), we can see that $ERROR_I$ (*Eq. 1.26*) is smaller than all the other. Meaning that, the Z500 cluster centroids are closer to the centroids of the rest of the parameters. So it would be best to represent each cluster with the "*Cluster Centroids of Z500*".

For $N < 10$ (choosing the leading <10 EOF modes), we can see that $ERROR_{II}$ (*Eq. 1.27*) and $ERROR_{III}$ (*Eq. 1.28*) are smaller than the others. Meaning that, the representatives "*Closest to Z500 centroids*" or "*Best fit to parameters centroids*", are closer to the centroids of the rest of the parameters. So it would be best to represent each cluster with those.

For $N = 20$ (choosing the leading 20 EOF modes), we can see that $ERROR_I$, $ERROR_{II}$ and $ERROR_{III}$ are about the same.

Regardless the choice of EOF modes, we can see that $ERROR_{IV}$ (*Eq. 1.29*) is greater than the rest. So it would not be advisable to represent the Z500 fields with "*Cluster Centroids of Z500*" and the rest of the parameters with the "*Closest to Z500 centroids*".

The more leading modes we choose from the EOF analysis, the better the "*Cluster Centroids of Z500*" describe the "mean weather scenario" (regarding all the given meteorological parameters). Clustering is performed on zonal-mean-removed monthly Z500 anomalies projected on the leading N EOF modes. In order to retain at least 70% of the variance, not only for the Z500 fields but for the rest of the meteorological parameters as well, N is taken equal to 20 or greater ($N \geq 20$).

Given the fact that $N > 20$ and the results of our test regarding $ERROR_{I-IV}$, the representatives "*Cluster Centroids of Z500*" or the "*Cluster centroids of each parameter*" would be the most appropriate to describe the "mean weather scenario".

2.4. Cluster analysis

The SOM algorithm was tested and showed analogous results with k-means (for the same number of clusters). The MATLAB script describing the way to perform SOM is presented in the Appendix 5: [MATLAB script for SOM].

In the present thesis we will present the results of the k-means technique regarding the cluster representatives for the cases "*Closest to Z500 centroids (EOF)"* and "*Cluster centroids of each parameter"* as they were described in section 1.2.6.

The "*Closest to Z500 centroids (EOF)*" representatives have the advantage of being an actual "weather scenario" that took place at some point in history. That means that all the meteorological parameters are more consistent with each other. Also, reconstructing the full patterns from the EOFs and PCs, leading to noise removal, can improve the results. For the month January they are presented in Table 4. The "*Cluster centroids of each parameter"* representatives are also chosen to be presented for the reasons mentioned in section 2.3. For the month January they are presented in Table 5.

We focus our interest in the period October-March as the winter season, since it is the period when most cyclones affect the Mediterranean Basin[5]. For those months, the "*Closest to Z500 centroids (EOF)*" and "*Cluster centroids of each parameter"* representatives are presented in the Appendix 3: [Cluster analysis].

Let us not forget that the patterns represent are monthly averaged fields. Having that in mind, the patterns are smoother compared to those originated from daily averaged or hourly data. So we do not anticipate to see neither extremely deep troughs nor cut-offs in Z500 or extremely low pressures with distinct centers in the surface (MSLP). The patterns should reflect an average of what happened during any given month. The same applies to all of the meteorological parameters examined.

In both cases of January (Table 4: "closest to Z500 centroids (EOF)" and Table 5: "cluster centroids of each parameter"), we see that all the examined parameters are consistent with each other though out all the atmospheric levels (upper and lower). Also, besides the different approach of calculating the cluster representatives, both cases show good resemblance with minor synoptic scale differences.

[5] HMSO, 1962

Table 4: January "weather scenarios". K-means based on Z500 observations for $\boldsymbol{k = 4}$. Cluster representatives of case "Closest to Z500 centroids (EOF)":

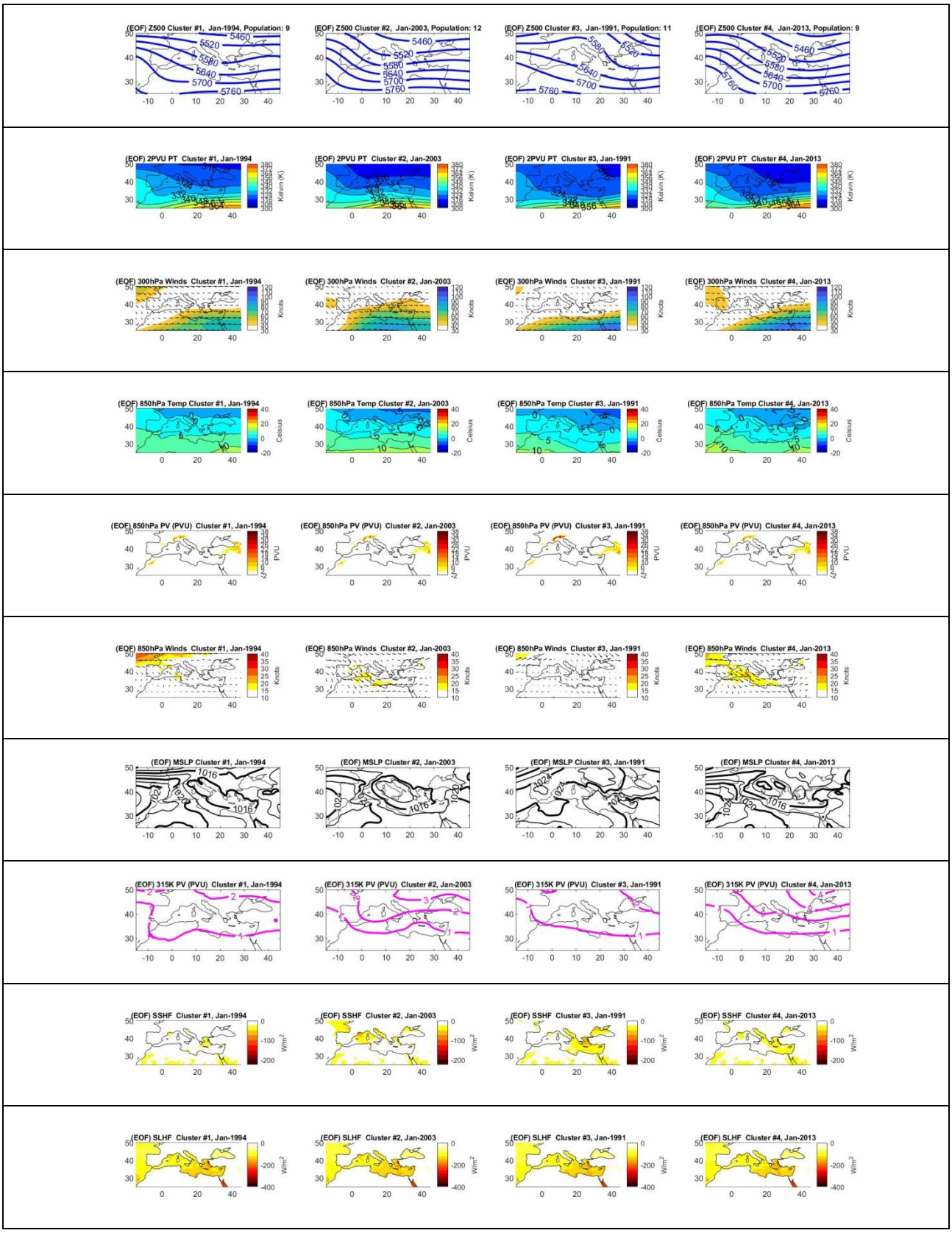
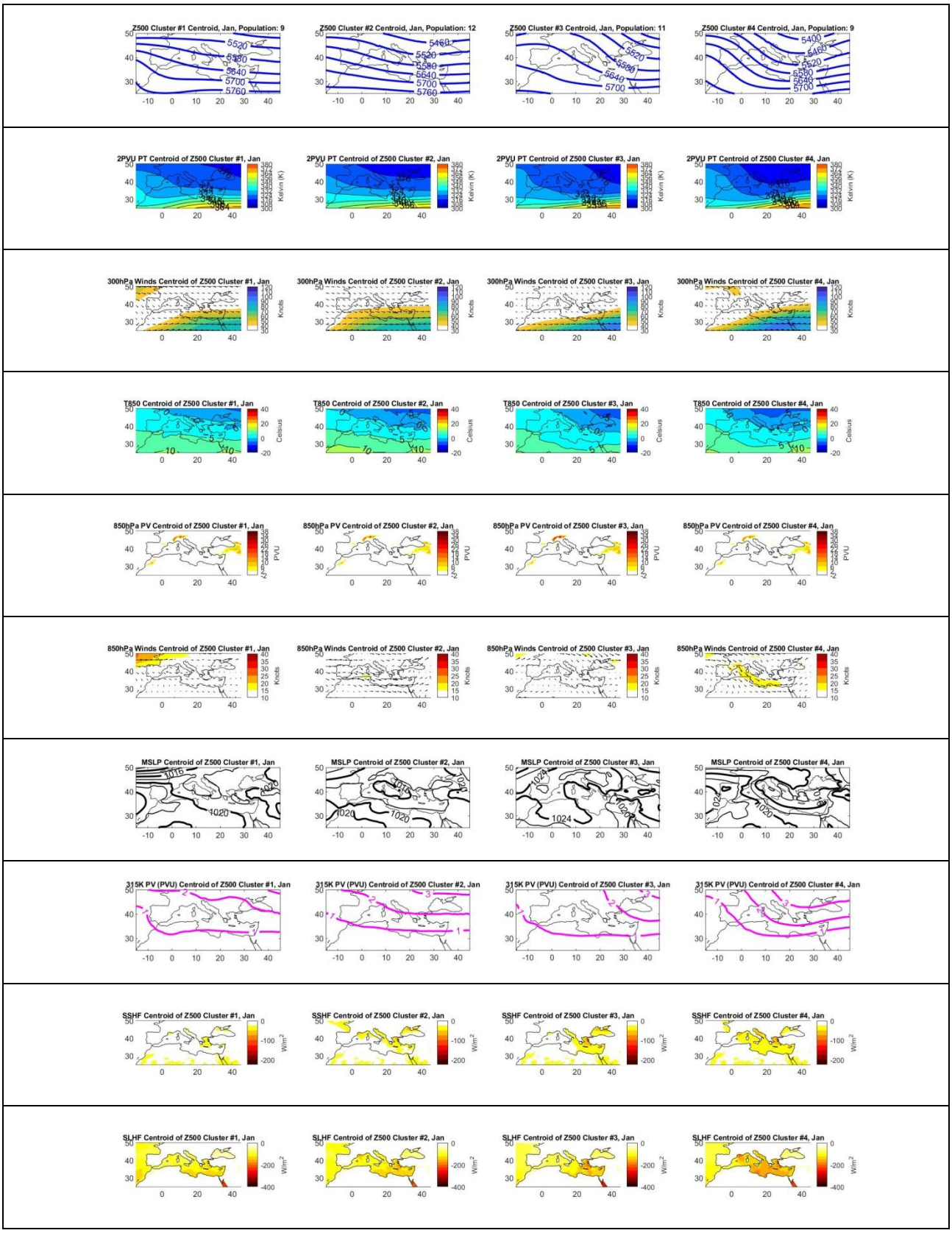


Table 5: January "weather scenarios". K-means based on Z500 observations for $k = 4$. Cluster representatives of case "Cluster centroids of each parameter"**:**



From the "weather scenarios" for the month January, analyzing each meteorological parameter separately, we can see the following:

- Z500:
  The 1st cluster representative shows a North West, almost zonal, flow over the Mediterranean. In the case of "closest to Z500 centroids (EOF)", there is a shallow trough over the Northern Mediterranean.

  The 2nd and 4th cluster representatives show a long wave through over the Mediterranean. In the 4th cluster the trough is deeper in the North (5400 gm).

  The 3rd cluster representative shows a long wave ridge affecting the Western Mediterranean and a trough effecting the Eastern Mediterranean and Black Sea.

  The 500hPa geopotential height is measured in geopotential meters (gm), with contour intervals of 60gm. The "population" in the title of each Z500 pattern, shows the number of members of each cluster.

  PT2PVU:
  The cluster representatives show low potential temperature air on the dynamic tropopause (2PVU surface) over the north. The potential temperature troughs are in complete correspondence with those of Z500. Also there is a strong potential temperature gradient towards the South, denoting where the subtropical jet stream maximums (jet streak) lie.

  The potential temperature on the 2PVU surface is measured in Kelvin (K), with contour intervals of 8K.

- W300:
  The 300hPa winds show the location of the jet stream. In all the cluster representatives, there is correspondence with the PT2PVU and also with the Z500 fields. The patterns show mainly the subtropical jet stream located in the south.

  The 300hPa wind speed is measured in knots, with color intervals of 10knots.

- T850:
  Here we can see the air temperature at the isobaric level of 850hPa. There are no large temperature gradients in those monthly patterns. There is correspondence with the Z500 cluster representatives, showing colder air temperature in the areas of low geopotential height. For the month of January, the 850hPa air temperature over the Mediterranean is 0-5^{o}C.

  The 850hPa temperature is measured in degrees Celsius (^{o}C), with contour intervals of 5^{o}C.

- PV850:
  Here we can see the areas of low level PV anomalies (on the 850hPa surface) on a monthly averaged basis. High values of potential vorticity (positive PV anomalies) are dominated over the Alps, as shown in all cluster representatives. This is due to intersection of the Alps with the 850hPa surface.

  The downward protrusion of high PV values cannot be seen in the monthly averaged patterns of the 850hPa level. Values higher than 1.5PVU are observed only over the Alps. Those patterns do not give us much information about the areas of cyclonic circulation on lower levels.

  The 850hPa potential vorticity is measured in PVU ($10^{-6} \cdot K \cdot Kg^{-1} \cdot m^2 \cdot s^{-1}$), with color intervals of 4PVU.

- W850:
  In the 1st, 2nd and 4th cluster representatives we can see a North-westerly flow entering the western Mediterranean accompanied by a flow separation due to the Alps. On the eastern parts of the Mediterranean, a westerly flow is gradually turning South-westerly as we move to the East.

  In the 3rd cluster representative, there is anticyclonic circulation in the western parts and a northerly flow over the central and eastern parts.

  Generally the monthly patterns show low wind speeds over the Mediterranean, with maximum values of 15-20knots in the first two cluster representatives. Those maximums form an arc that starts from the Gulf of Lion, to Tunisia and over the northern coasts of Libya and Egypt.

  The 850hPa wind speed is measured in knots, with color intervals of 5knots.

- MSLP:
  In the 1st, 2nd and 4th cluster representatives we can see that cyclonic circulation and low pressures are found in the areas of the Gulf o Genoa, the Tyrrhenian Sea, the southern Italy, the Adriatic Sea, the Ionian Sea, the Aegean Sea and in the vicinity of Crete near Cyprus.

  In the 3rd cluster representative, low pressures are found outside the Mediterranean Sea manly over the North East Black Sea.

  Generally the monthly patterns do not show extremely low pressures. The minimum pressure shown in the patterns is 1014hPa.

  The mean sea level pressure is measured in hPa, with contour intervals of 2hPa (thin lines) and 4hPa (thick lines).

- PV315:
  In the cluster representatives we can see the PV field on the 315K isentropic surface. When stratospheric air protrudes downward into the troposphere, an upper level PV anomaly develops. PV of magnitude larger than 2PVU denotes stratospheric protrusion and cyclonic circulation in the upper levels.

  The dynamical tropopause (2PVU) in the 1st, 2nd and 4th cluster representatives is located over the northern parts of the Mediterranean and PV increases toward the poles. Those areas of cyclonic PV anomalies are accompanied by upper-level troughs, consistent with both Z500 and PT2PVU fields.

  In the 3rd cluster representative, larger than 2PVU is found in the North East, mainly over the Black Sea.

  When a positive upper level PV anomaly is being advected over a zone of strong, low-level temperature gradient, a rotation is induced in the lower levels of the baroclinic zone, which causes warm advection at these levels and intensifies the cyclonic vortex there.

  The 315K potential vorticity is measured in PVU ($10^{-6} \cdot K \cdot Kg^{-1} \cdot m^2 \cdot s^{-1}$), with contour intervals of 1PVU.

- SSHF & SLHF:
  As already mentioned in section 1.1.4, sensible and latent heat fluxes are the main mechanisms that contribute significantly to energy enrichment of the lower tropospheric layers. Subsequently, they provide energy for depressions.

  During winter, depressions are formed over the various cyclogenesis regions of the western and central Mediterranean Sea. These depressions move eastwards, and being reinforced by the warm Mediterranean water on a limited time scale but also on climatological time scales.

  In the 1st, 2nd and 4th cluster representatives, SSHF is lowest (-50 to -75 W/m^2) over the northern coasts of the Mediterranean.

  In the 3rd cluster representative, the SSHF is lowest (-50 to -75 W/m^2) over the Aegean Sea and the northern Black Sea.

  The SLHF cluster representatives show lower values mainly over the southern Mediterranean, the Gulf of Lion and the Aegean Sea.

  The ECMWF convention for vertical fluxes is positive downwards. So the larger the negative value the more heat is transferred from the Earth's surface to the atmosphere through the effects of turbulent air motion. For the month of January, the heat exchange mostly takes place over the warmer Mediterranean Sea for both SSHF and SLHF.

  The SSHF and SLHF are measured in W/m^2 , with color intervals of 25 W/m^2.

In the winter, the MB offers all the cyclogenetic factors, being highly baroclinic, warmer than its surroundings and positioned at the lee of mountain ridges, such as the Atlas ridge in Morocco, the European Alps and the Taurus ridge in Turkey[6]. The MB winter cyclones have some special characteristics:

- they develop mainly at the northern Mediterranean coasts.
- their horizontal size varies between meso-scale and synoptic-scale[7].
- their life span is around one to four days.
- their typical speed is $\sim 5\ ^{m}/_{s}$.
- they reach the tropopause[8].

When evaluating a monthly pattern, surface areas with low-pressure cyclonic flow accompanied by cyclogenetic factors in the upper and lower atmospheric levels, show evidence of the locations where cyclones form, move and die on a monthly basis. They show us the main areas affected by cyclonic activity during the month we examine.

Many efforts have been made to map the MB cyclogenetic regions[9]. The main regions are: the Gulf of Genoa, the Iberian Peninsula, southern Italy, the Aegean Sea, the Tyrrhenian Sea, south of Sardinia, the Ionian Sea, the Baleares Islands and Cyprus.

In the previous pages we presented an interpretation of the results for the month January. The same can be done for the other winter months (Appendix 3: [Cluster analysis]). We are looking for areas of low pressure cyclonic circulation in the MSLP patterns, under a 500hPa geopotential through, under a potential temperature trough on the 2PVU level, under cyclonic circulation on the 850hPa level, under positive values of potential vorticity on the 315K isentropic surface and with low values of SSHF and SLHF. Analyzing those results of the monthly “weather scenarios”, we were able to re-identify the areas of the MB that are most affected by cyclones. Those are:

- Over the Aegean Sea and or the lee of Taurus Mountains.
- Mostly in the Gulf of Genoa or the lee of the Pyrenees
- Mostly south of the Italian Alps.
- Mostly in the vicinity of Crete (and near Cyprus).
- In the areas of Sardinia, Sicily and north Libya.
- Mostly next to the Italian peninsula.

[6] HMSO, 1962; Alpert et al., 1990, 1996; Trigo et al., 1999; Lionello et al., 2006
[7] Alpert et al., 1990; Trigo et al., 1999; Campins et al., 2000; Trigo, 2006; Lionello et al., 2006
[8] Alpert and Ziv, 1989
[9] Petterssen, 1956; Alpert et al., 1990; Trigo et al., 1999; Campins et al., 2000; Maheras et al., 2001; Genoves´ et al., 2006; Lionello et al., 2006; Trigo, 2006

3. Conclusion

The EOF alone can be used as a method for pattern extraction. The leading EOF modes indicate the most prevailing anomalies of each meteorological parameter over the Mediterranean. This method has been extensively used in the ocean and atmospheric sciences to extract deferent dominant patterns and to separate dimensions of complex systems. In our case, the method is used as a data reduction tool in order to efficiently apply a clustering algorithm for pattern extraction.

The SVD of coupled fields was used in order to reveal the correlations between the meteorological parameters that we used in this thesis. Even though it was known beforehand that they highly affect each other and have big impact on each other, the results verified the hypothesis. From the SVD maps, the modes of behavior in which the fields variations are strongly coupled reveal how the two fields interact.

For the spatio-temporal data mining, the k-means clustering is used as a basis. Also SOM (Self Organizing Map) was tested for verification purposes of the quality of the k-means clustering. Centroids and cluster representatives were calculated in order to extract patterns of monthly "weather scenarios". Also some distance metrics, were introduced in order to optimize the performance and better suit the characteristics of our dataset. Those monthly "weather scenarios", exhibit the areas of the Mediterranean that are more affected by surface cyclogenesis during winter. Those are:

- Over the Aegean Sea and or the lee of Taurus Mountains.
- Mostly in the Gulf of Genoa or the lee of the Pyrenees
- Mostly south of the Italian Alps.
- Mostly in the vicinity of Crete (and near Cyprus).
- In the areas of Sardinia, Sicily and north Libya.
- Mostly next to the Italian peninsula.

Other unsupervised machine learning techniques can be tested for comparison such as Hierarchical Clustering, k-medioids etc. Also, the labeled dataset could be used to train and test a supervised machine learning technique such as CNN (Convolutional neural network) or Log-Reg (Logistic-Regression). The CNN or Log-Reg can be used to predict which cluster index a Z500 pattern will belong to in the future. Deep learning techniques such as convolutional neural networks (CNNs) can potentially provide powerful tools for classifying, identifying, and predicting patterns in climate and environmental data. However, because of the inherent complexities of such data, which are often spatio-temporal, chaotic, and non-stationary, the CNN algorithms must be designed/evaluated for each specifc dataset and application. Yet CNN, being a supervised technique, requires a large labeled dataset to start. In our case, the monthly dataset is not large enough to train such a machine learning technique. It would have a meaning if daily patterns were examined and the size of our samples was measured in thousands. In the 41 year period of our case, only 492 monthly observations account for each meteorological parameter. For the reasons mentioned, presenting and evaluating a supervised machine learning technique is outside the scope of this thesis.

Reference list

[1] [Alpert P. and Ziv B.] (1989): “The Sharav cyclone: observation and some theoretical considerations”. J. Geoghys. Res., 94, 18 495–18 514, 1989.

[2] [Alpert P., Neeman B. U. and Shay-El Y.] (1990): “Climatological Analysis of Mediterranean Cyclones Using ECMWF Data”. Tellus, 42A, 65–77.

[3] [Bao, M., and J. Wallace] (2015): “Cluster analysis of Northern Hemisphere wintertime 500hPa flow regimes 1920-2014”. Journal of the atmospheric sciences.

[4] [Campins J., Genoves A., Jansa A., Guijarro J. A. and Ramis C.] (2000): “A Catalogue and a Classification of Surface Cyclones for the Western Mediterranean”. Int. J. Climatol., 20, 969–984.

[5] [Cayan D. R.] (1992a): “Latent and sensible heat flux anomalies over the northern oceans: the connection to monthly atmospheric circulation”. Journal of Climate 5: 354–369.

[6] [Cayan D. R.] (1992b): “Latent and sensible heat flux anomalies over the northern oceans: driving the sea surface temperature”. Journal of Physical Oceanography 22: 859–881.

[7] [Chattopadhyay A., Hassanzadeh P., Pasha S.] (2020): “Predicting clustered weather patterns: A test case for applications of convolutional neural networks to spatio-temporal climate data”.

[8] [Danielsen E. F.] (1968): “Stratospheric-Tropospheric Exchange Based on Radioactivity, Ozone and Potential Vorticity”. Journal of Atmospheric Sciences. Volume 25: Issue 3.

[9] [Flocas H. A. and Karacostas T. S] (1996): “Cyclogenesis over the Aegean Sea: Identification and synoptic categories”. Department of Meteorology and Climatology, Aristotelian University of Tbessaloniki.

[10] [Genoves A., Campins J. and Jans A.] (2006): “Intense Storms in the ` Mediterranean: A First Description from the ERA-40 Perspective”. Adv. Geosci., 7, 163–168.

[11] [Hannachi A.] (2005): “Pattern hunting in climate: a new method for finding trends in gridded climate data”. International Journal of climatology.

[12] [Hersbach, H., Bell, B., Berrisford, P., Biavati, G., Horányi, A., Muñoz Sabater, J., Nicolas, J., Peubey, C., Radu, R., Rozum, I., Schepers, D., Simmons, A., Soci, C., Dee, D., Thépaut, J-N.] (2019): ERA5 monthly averaged data on single levels from 1979 to present. Copernicus Climate Change Service (C3S) Climate Data Store (CDS).

[13] [Hersbach, H., Bell, B., Berrisford, P., Biavati, G., Horányi, A., Muñoz Sabater, J., Nicolas, J., Peubey, C., Radu, R., Rozum, I., Schepers, D., Simmons, A., Soci, C., Dee, D., Thépaut, J-N.] (2019): ERA5 monthly averaged data on pressure levels from 1979 to present. Copernicus Climate Change Service (C3S) Climate Data Store (CDS).

[14] [HMSO] (1962): “Weather in the Mediterranean I: General Meteorology”. 2d ed., Her Majesty’s Stationery Office, 362 p.

[15] [Holton J. R., Haynes P. H., McIntyre M. E., Douglass A. R.] (1995): “Stratosphere-Troposphere Exchange”. Reviews of Geophysics - REV GEOPHYS. 33. 10.1029/95RG02097.

[16] [Kohonen T.] (1982): "Self-Organized Formation of Topologically Correct Feature Maps". Biological Cybernetics 43, 59-69.

[17] [Lionello P., Malanbotte-Rizzoli P. and Boscolo R. (Eds.)] (2006): “Mediterranean Climate Variability, Developments in Earth and Environmental Sciences 4”. Elsevier, pp. 325–372.

[18] [Lolis C. J., Bartzokas A.] (2004): “Relation between Sensible and Latent heat fluxes in the Mediterranean and Precipitation in the Greek area during winter”.

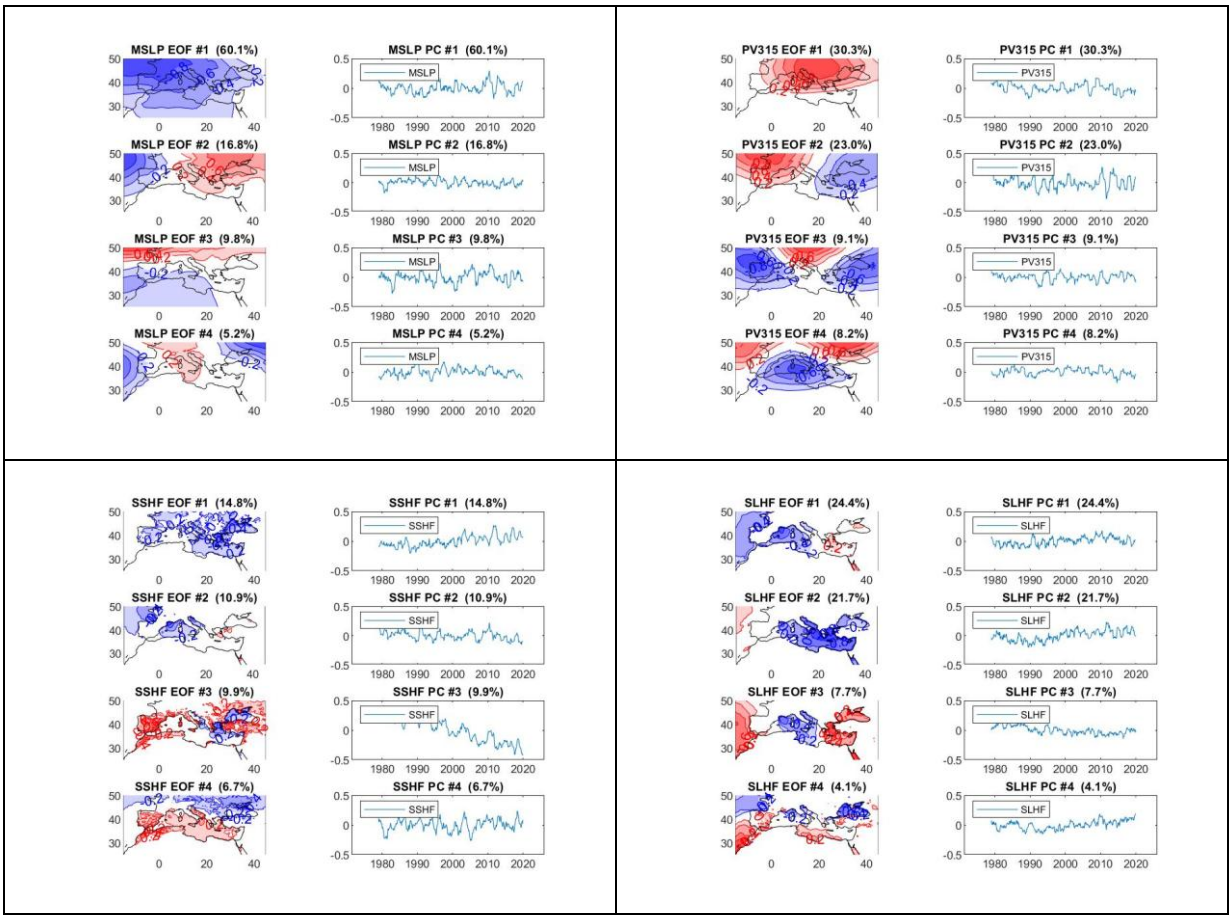
[19] [MacQueen, J. B.] (1967): “Some Methods for classification and Analysis of Multivariate Observations”. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press. pp. 281–297.

[20] [Maheras P., Flocas H., Patrikas I. and Anagnostopoulou Ch.] (2001): “A 40 year objective climatology of surface cyclones in the Mediterranean region: spatial and temporal distribution”. Int. J. Climatol., 21, 109–130.

[21] [McIntyre M.] (1982): “The use of potential vorticity and low-level temperature/moisture to understand extratropical cyclogenesis”.
[22] [Metaxas D. A.] (1978): “Evidence on the importance of diabatic heating as a divergence factor in the Mediterranean”. Archiv fur Meteorologie, Geophysik und Bioklimatologie, Series A 27: 69–80.
[23] [Michelangeli P. A, Vautard R., Lergas B.] (1995) “Weather Regimes: Recurrence and Quasi Stationarity”. J. Atmos. Sci. 52, 1237–1256.
[24] [Mohamad I. B., Usman D.] (2013): “Standardization and Its Effects on K-Means”. Research Journal of Applied Sciences, Engineering and Technology 6(17): 3299-3303.
[25] [Moron V., Robertson A. W., Quan J-H, Ghil M.] (2014): “Weather types across the Maritime Continent: From the diurnal cycle to interannual variations”.
[26] [Petterssen S.] (1990): “Weather Analysis and Forecasting”. Vol I., McGrawHill, New York, 428.
[27] [Preisendorfer, R. W.] (1988): “Principal Component Analysis in Meteorology and Oceanography”. Elsevier.
[28] [Repapis C. C., Metaxas D. A., Zerefos C. S.] (1978): “A contribution to climatology of sensible heat flux over the Mediterranean Sea”. Bulletin of the Hellenic Meteorological Society 3(2): 1–20.
[29] [Riddle E. E., Stoner M. B., Johnson N. C., L’Heureux M. L., Colins D. C., Felbstein S. B.] (2012): “The impact of the MJO on clusters of wintertime circulation anomalies over the North American region”. Clim Dyn 40: 1749-1766.
[30] [Romen M., Zin B., Saaroni H.] (2007): “Scenarios in the development of Mediterranean cyclones”. Adv. Geosci., 12, 59-65.
[31] [Sciremammano, F.] (1979): “A suggestion for the presentation of correlations and their significance levels”. J. Phys. Oceanogr.
[32] [Sheshadri A., Plumb R. A.] (2017): “Propagating Annular Modes: Empirical Orthogonal Functions, Principal Oscillation Patterns, and Time Scales”. Journal of the atmospheric sciences.
[33] [Storch von H. and Navara, A.] (1995): “Analyses of Climate Variability – Applications of Statistical Techniques”. Springer.
[34] [Trigo I. F, Davies T. D. and Bigg G. R.] (1999): “Objective Climatology of Cyclones in the Mediterranean Region”. J. Climate, 12, 6, 1685– 1696.
[35] [Trigo I. F.] (2006): “Climatology and Interannual Variability of StormTracks in the Euro-Atlantic Sector: a Comparison between ERA40 and NCEP/NCAR Reanalysis”. Clim. Dynam., 26(2–3), 127– 143.
[36] [Trigo I. F., Bigg G. R., Danies T. D.] (2002): “Climatology of Cyclogenesis Mechanisms in the Mediterranean”.
[37] [Venegas S. A., Bjomsson H.] (1997): “A manual for EOF and SVD analyses of Climatic Data”.
[38] [Venegas S. A., Mysak L. A., and Straub D. N.] (1996): “Evidence for interannual and interdecadal climate variability in the South Atlantic”. Geophys. Res. Lett.
[39] [Wallace J. M., Smith C., Bretherton C. S.] (1992): “Singular Value Decomposition of wintertime sea surface temperature and 500-mb height anomalies”. J. Clim.
[40] [Zhong S., Yu L., Heilman W. E., Bian X., Fromm H.] (2020): “Synoptic weather patterns for large wildfires in the northwestern United States—a climatological analysis using three classification methods”.

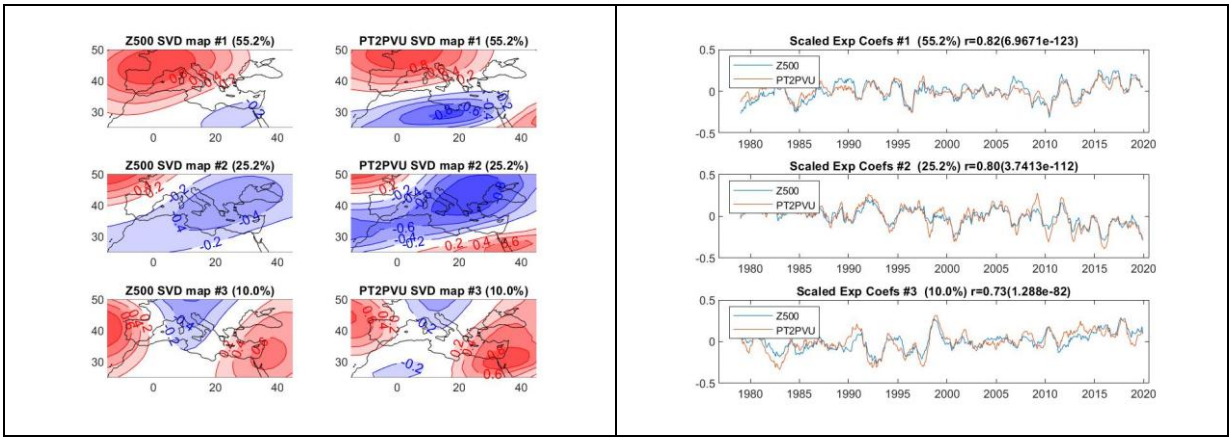
Appendix 1: [EOF]

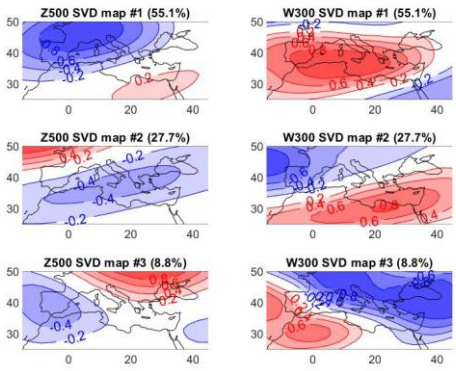
Table 6: Spatial patterns (EOF maps) and time series (PCs) of the leading four EOF modes for all the variables (fields of anomalies of Z500, PT2PVU, W300, T850, PV850, W850, MSLP, PV315, SSHF, SLHF). Spatial patterns are presented as dimensionless maps with amplitudes scaled to range [-1, 1]. Negative contours are blue. Contour interval is 0.2. Time series are smoothed by a 13-month running mean and amplitudes are scaled to range [-1, 1].

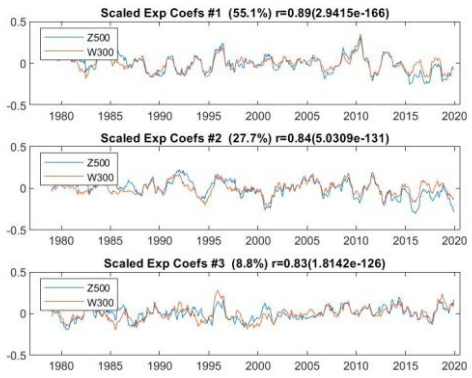


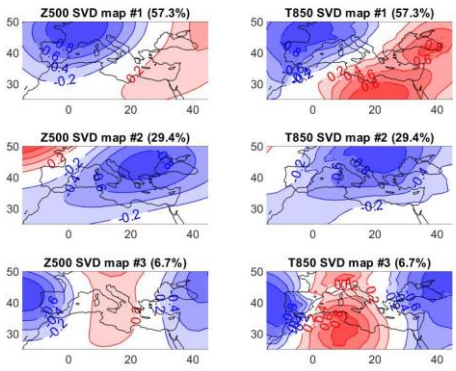
Appendix 2: [SVD]

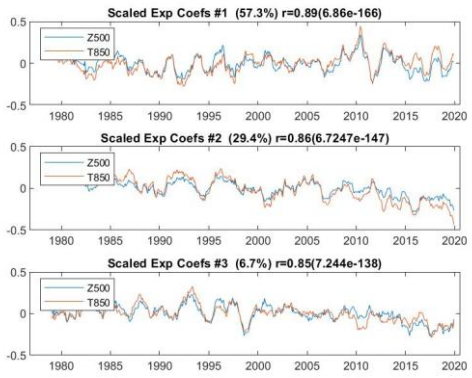
Table 7: Spatial patterns (SVD maps, left column) and time series (PCs, right column) of the leading three SVD modes for the coupled fields of Z500 fields of anomalies, with each of the rest variables (fields of anomalies of PT2PVU, W300, T850, PV850, W850, MSLP, PV315, SSHF, SLHF). Spatial patterns are presented as dimensionless maps with amplitudes scaled to range [-1, 1]. Negative contours are blue. Contour interval is 0.2. Time series (expansion coefficients or PCs) are smoothed by a 13-month running mean and amplitudes are scaled to range [-1, 1]. Blue lines are the Z500 PCs.

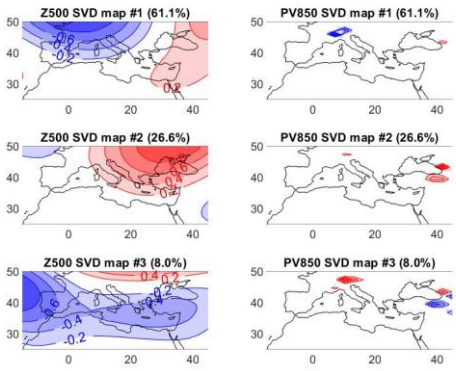


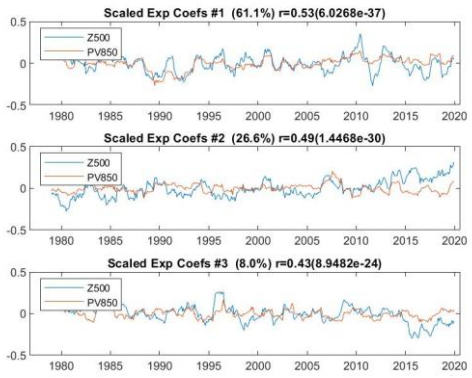

Z500 SVD map #1 (55.1%)
W300 SVD map #1 (55.1%)
Z500 SVD map #2 (27.7%)
W300 SVD map #2 (27.7%)
Z500 SVD map #3 (8.8%)
W300 SVD map #3 (8.8%)

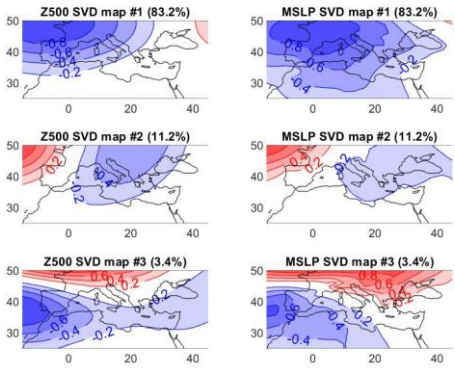

Scaled Exp Coefs #1 (55.1%) r=0.89(2.9415e-166)
Scaled Exp Coefs #2 (27.7%) r=0.84(5.0309e-131)
Scaled Exp Coefs #3 (8.8%) r=0.83(1.8142e-126)
Z500
W300

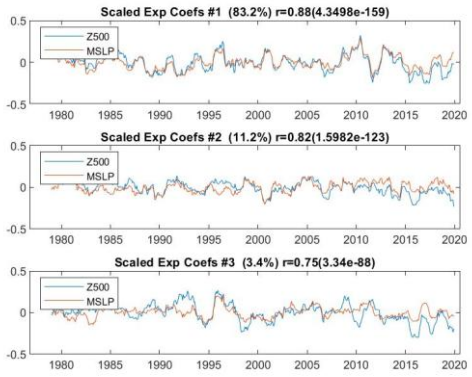

Z500 SVD map #1 (57.3%)
T850 SVD map #1 (57.3%)
Z500 SVD map #2 (29.4%)
T850 SVD map #2 (29.4%)
Z500 SVD map #3 (6.7%)
T850 SVD map #3 (6.7%)

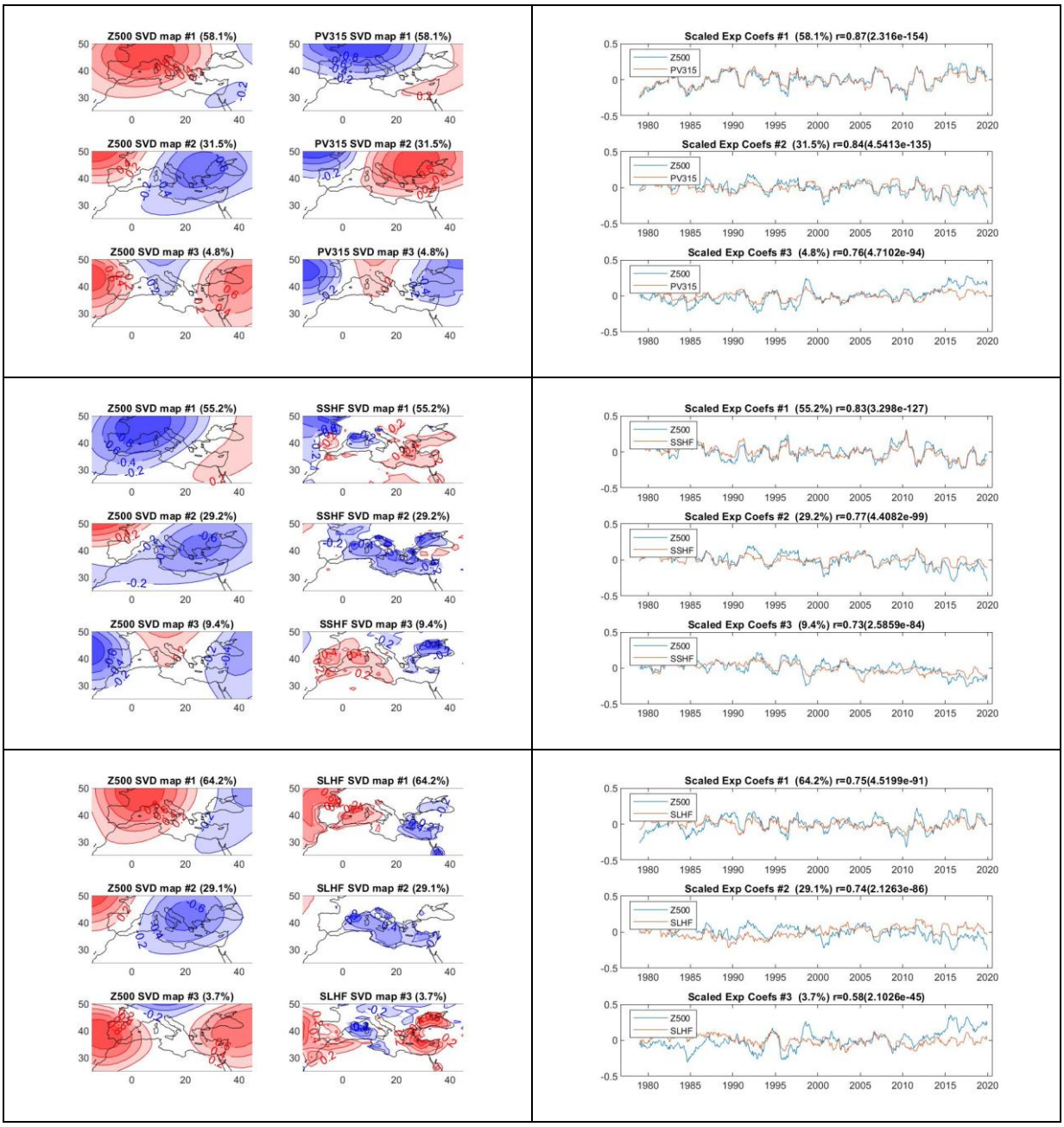

Scaled Exp Coefs #1 (57.3%) r=0.89(6.86e-166)
Scaled Exp Coefs #2 (29.4%) r=0.86(6.7247e-147)
Scaled Exp Coefs #3 (6.7%) r=0.85(7.244e-138)
Z500
T850


Z500 SVD map #1 (61.1%)
PV850 SVD map #1 (61.1%)
Z500 SVD map #2 (26.6%)
PV850 SVD map #2 (26.6%)
Z500 SVD map #3 (8.0%)
PV850 SVD map #3 (8.0%)


Scaled Exp Coefs #1 (61.1%) r=0.53(6.0268e-37)
Scaled Exp Coefs #2 (26.6%) r=0.49(1.4468e-30)
Scaled Exp Coefs #3 (8.0%) r=0.43(8.9482e-24)
Z500
PV850


Z500 SVD map #1 (83.2%)
MSLP SVD map #1 (83.2%)
Z500 SVD map #2 (11.2%)
MSLP SVD map #2 (11.2%)
Z500 SVD map #3 (3.4%)
MSLP SVD map #3 (3.4%)


Scaled Exp Coefs #1 (83.2%) r=0.88(4.3498e-159)
Scaled Exp Coefs #2 (11.2%) r=0.82(1.5982e-123)
Scaled Exp Coefs #3 (3.4%) r=0.75(3.34e-88)
Z500
MSLP


Z500 SVD map #1 (58.1%)
PV315 SVD map #1 (58.1%)
Z500 SVD map #2 (31.5%)
PV315 SVD map #2 (31.5%)
Z500 SVD map #3 (4.8%)
PV315 SVD map #3 (4.8%)
Scaled Exp Coefs #1 (58.1%) r=0.87(2.316e-154)
Scaled Exp Coefs #2 (31.5%) r=0.84(4.5413e-135)
Scaled Exp Coefs #3 (4.8%) r=0.76(4.7102e-94)
Z500
PV315
Z500 SVD map #1 (55.2%)
SSHF SVD map #1 (55.2%)
Z500 SVD map #2 (29.2%)
SSHF SVD map #2 (29.2%)
Z500 SVD map #3 (9.4%)
SSHF SVD map #3 (9.4%)
Scaled Exp Coefs #1 (55.2%) r=0.83(3.298e-127)
Scaled Exp Coefs #2 (29.2%) r=0.77(4.4082e-99)
Scaled Exp Coefs #3 (9.4%) r=0.73(2.5859e-84)
Z500
SSHF
Z500 SVD map #1 (64.2%)
SLHF SVD map #1 (64.2%)
Z500 SVD map #2 (29.1%)
SLHF SVD map #2 (29.1%)
Z500 SVD map #3 (3.7%)
SLHF SVD map #3 (3.7%)
Scaled Exp Coefs #1 (64.2%) r=0.75(4.5199e-91)
Scaled Exp Coefs #2 (29.1%) r=0.74(2.1263e-86)
Scaled Exp Coefs #3 (3.7%) r=0.58(2.1026e-45)
Z500
SLHF

Appendix 3: [Cluster analysis]

Table 8: October "weather scenarios". K-means based on Z500 observations for $k = 4$. Cluster representatives of case "Closest to Z500 centroids (EOF)":

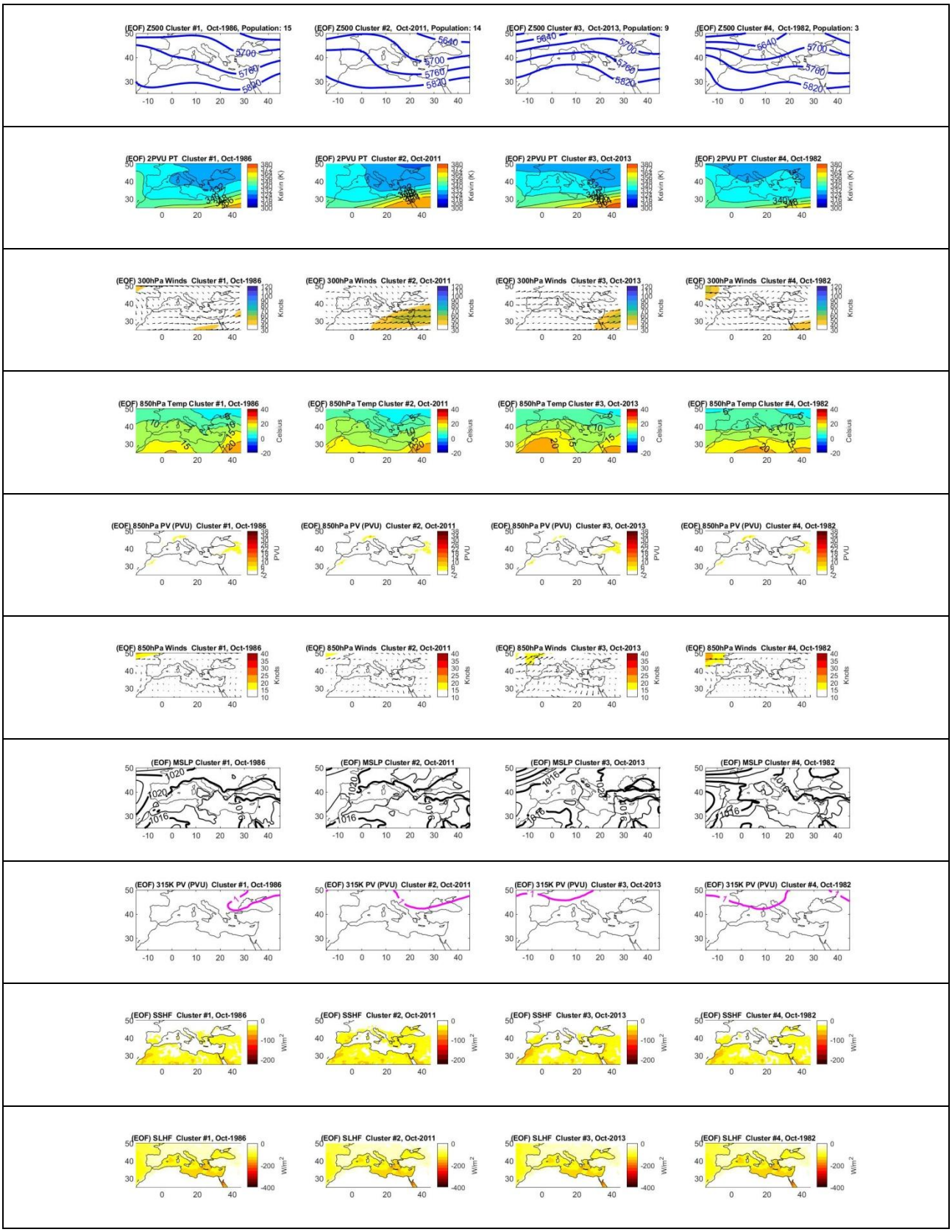


Table 9: October "weather scenarios". K-means based on Z500 observations for $k = 4$. Cluster representatives of case "Cluster centroids of each parameter":

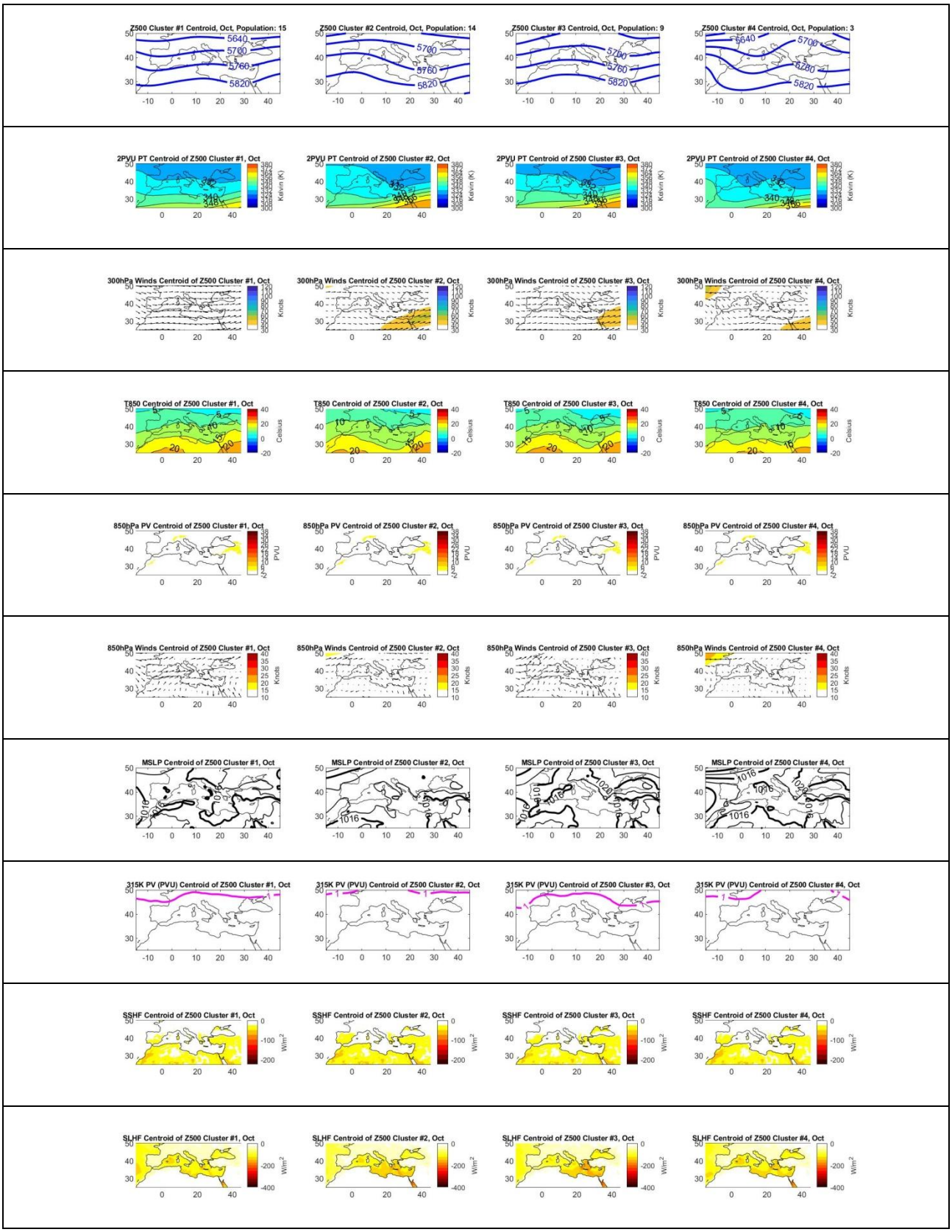


Table 10: November "weather scenarios". K-means based on Z500 observations for $k = 4$. Cluster representatives of case "Closest to Z500 centroids (EOF)":

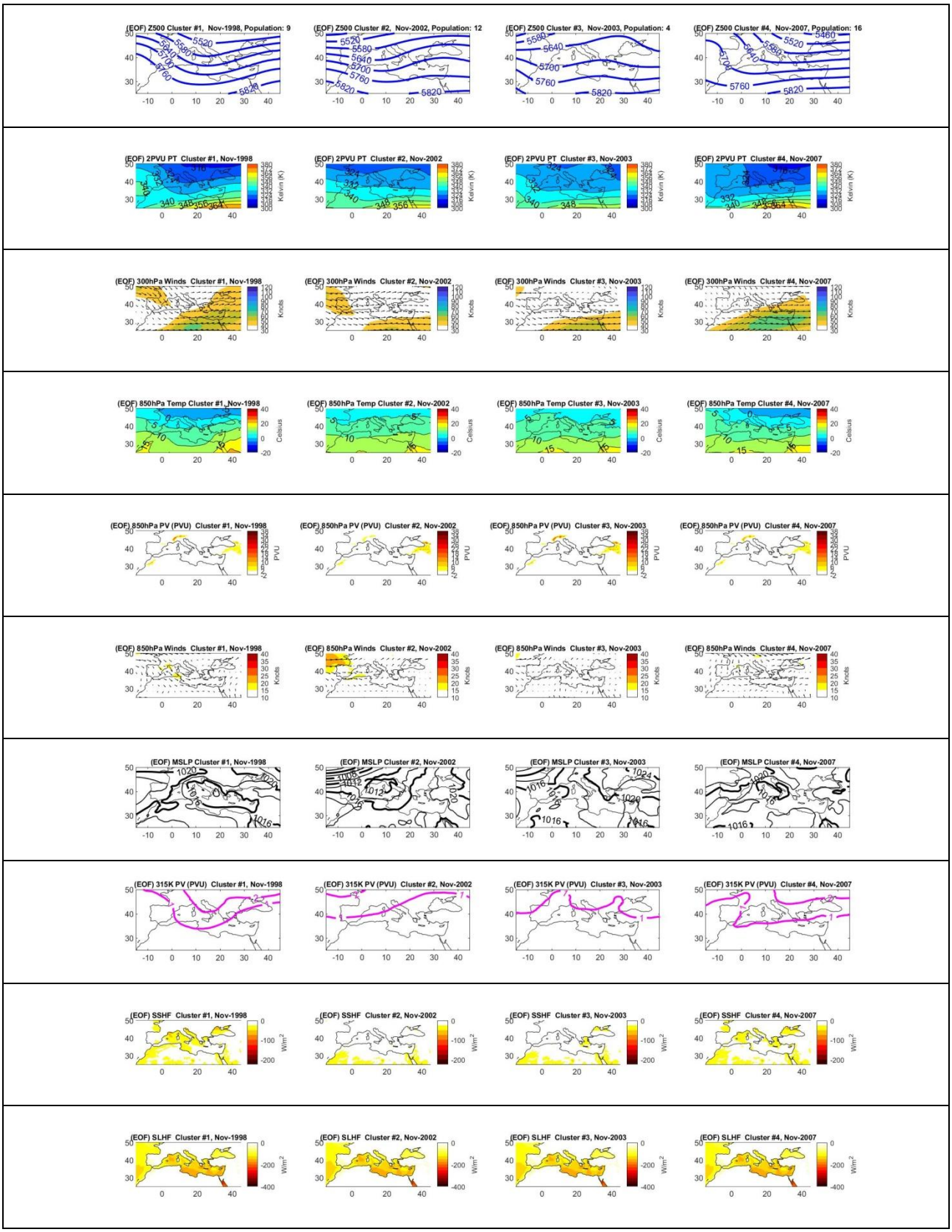


Table 11: November "weather scenarios". K-means based on Z500 observations for $\boldsymbol{k = 4}$. Cluster representatives of case "Cluster centroids of each parameter"**:**

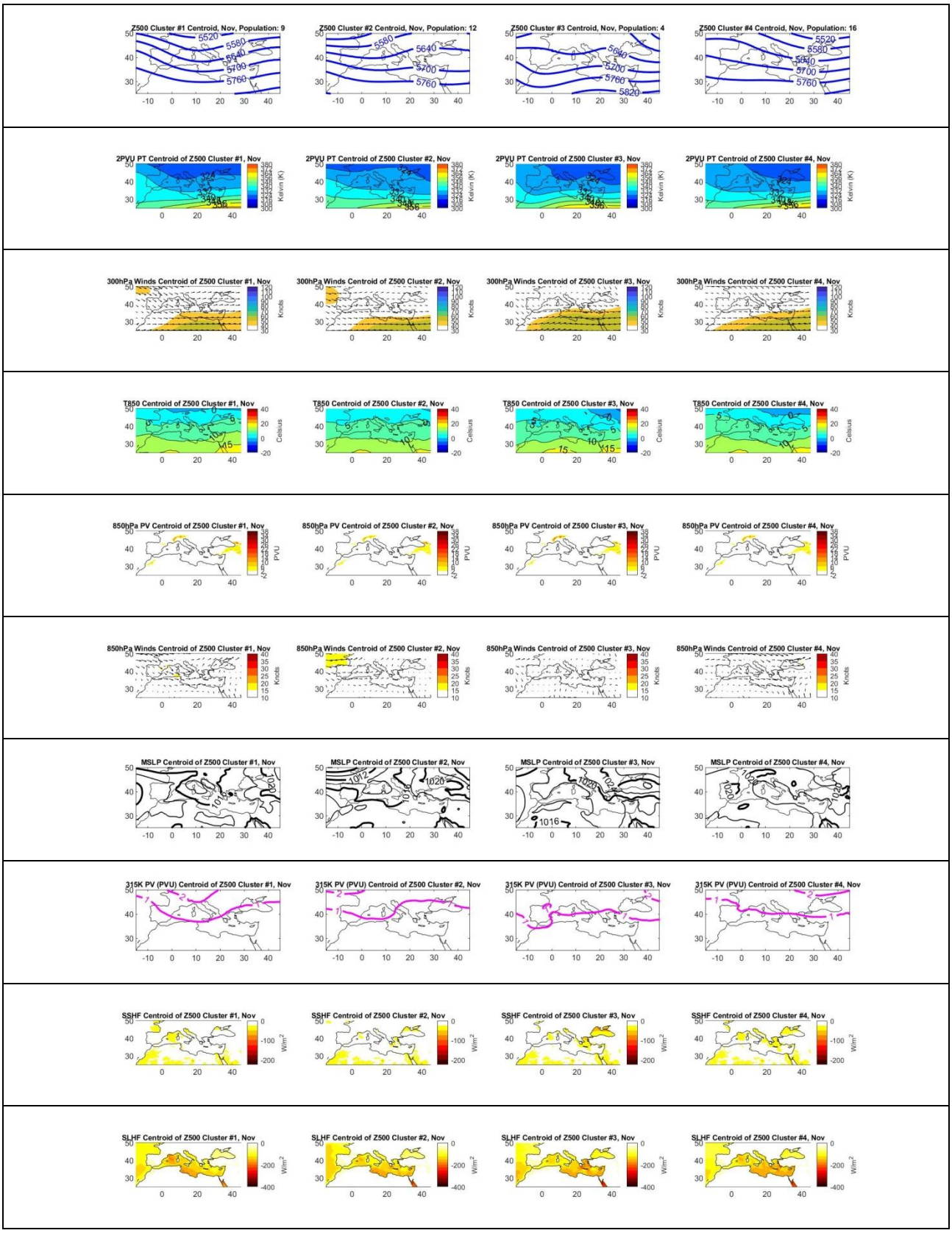


Table 12: December "weather scenarios". K-means based on Z500 observations for $k = 4$. Cluster representatives of case "Closest to Z500 centroids (EOF)":

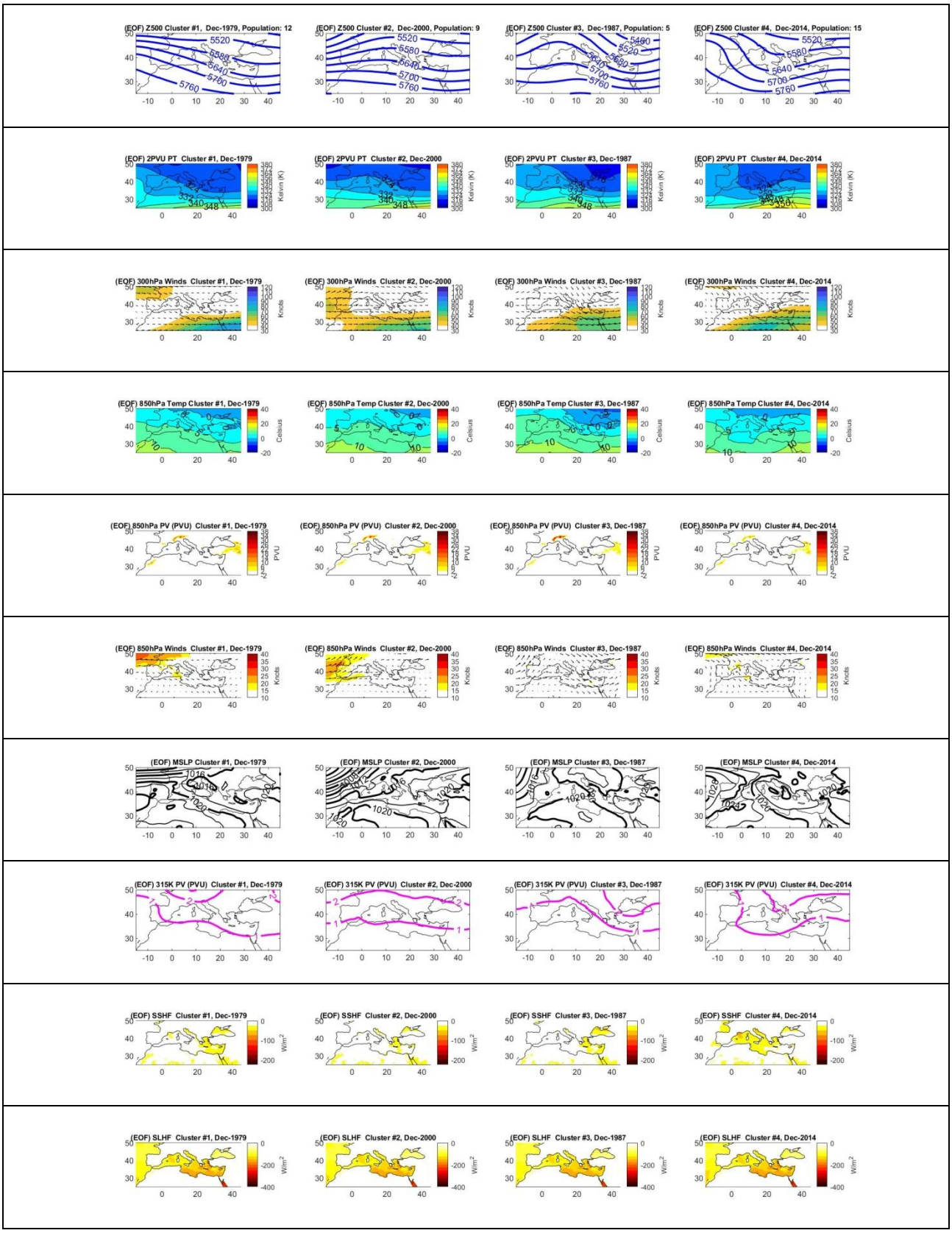


Table 13: December "weather scenarios". K-means based on Z500 observations for $\boldsymbol{k = 4}$. Cluster representatives of case "Cluster centroids of each parameter"**:**

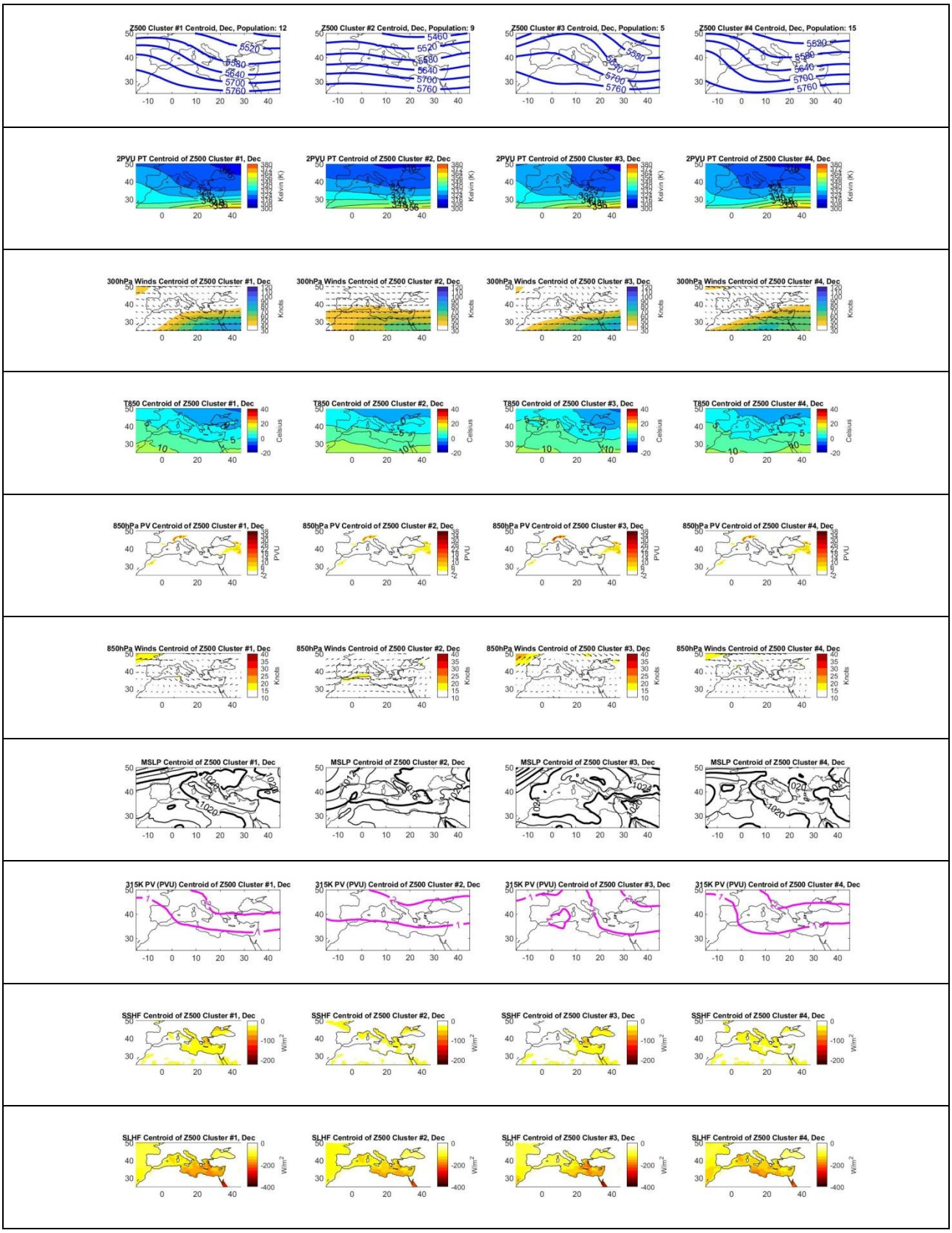


Table 14: January "weather scenarios". K-means based on Z500 observations for $k = 4$. Cluster representatives of case "Closest to Z500 centroids (EOF)":

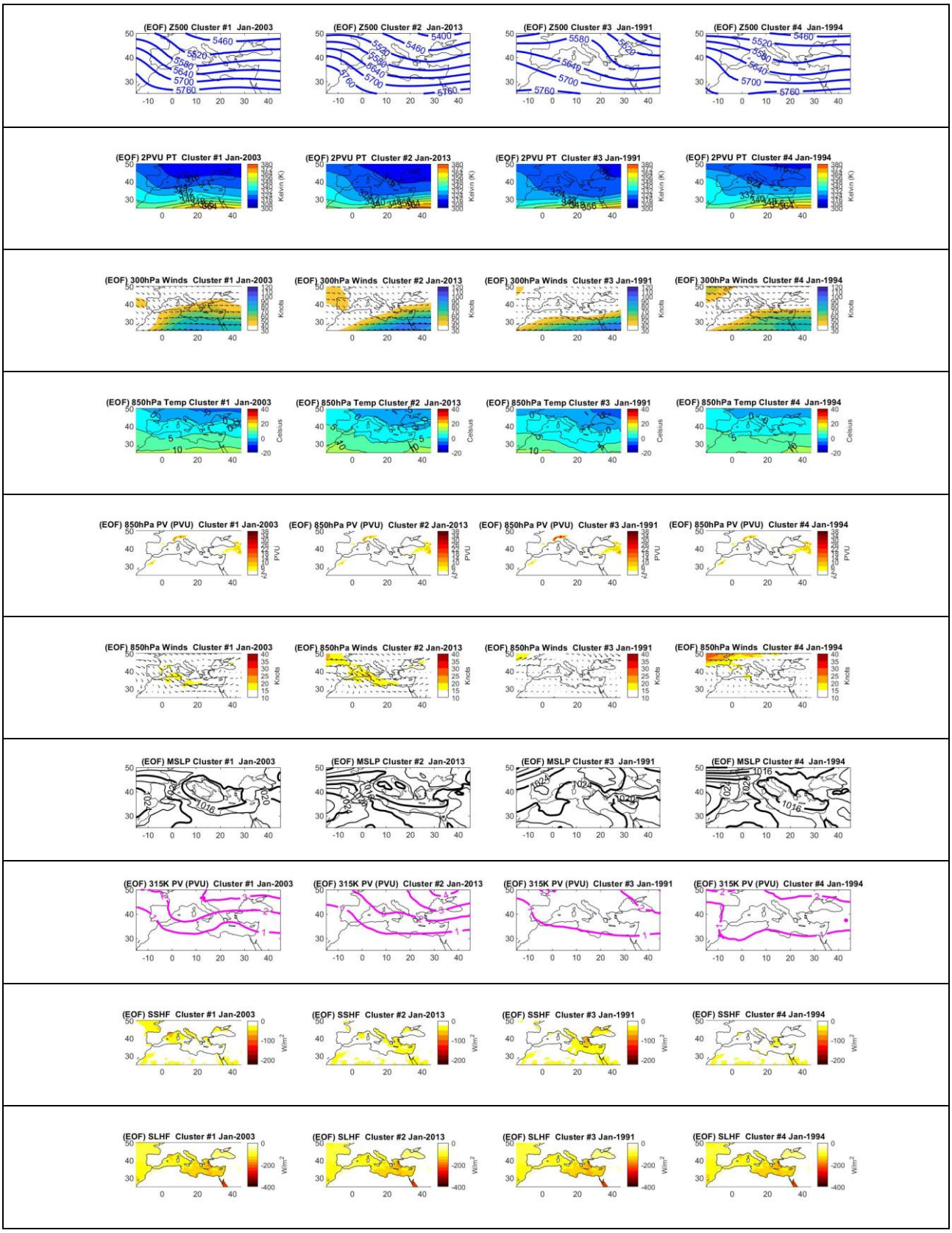


Table 15: January "weather scenarios". K-means based on Z500 observations for $\boldsymbol{k = 4}$. Cluster representatives of case "Closest to Z500 centroids (EOF)":

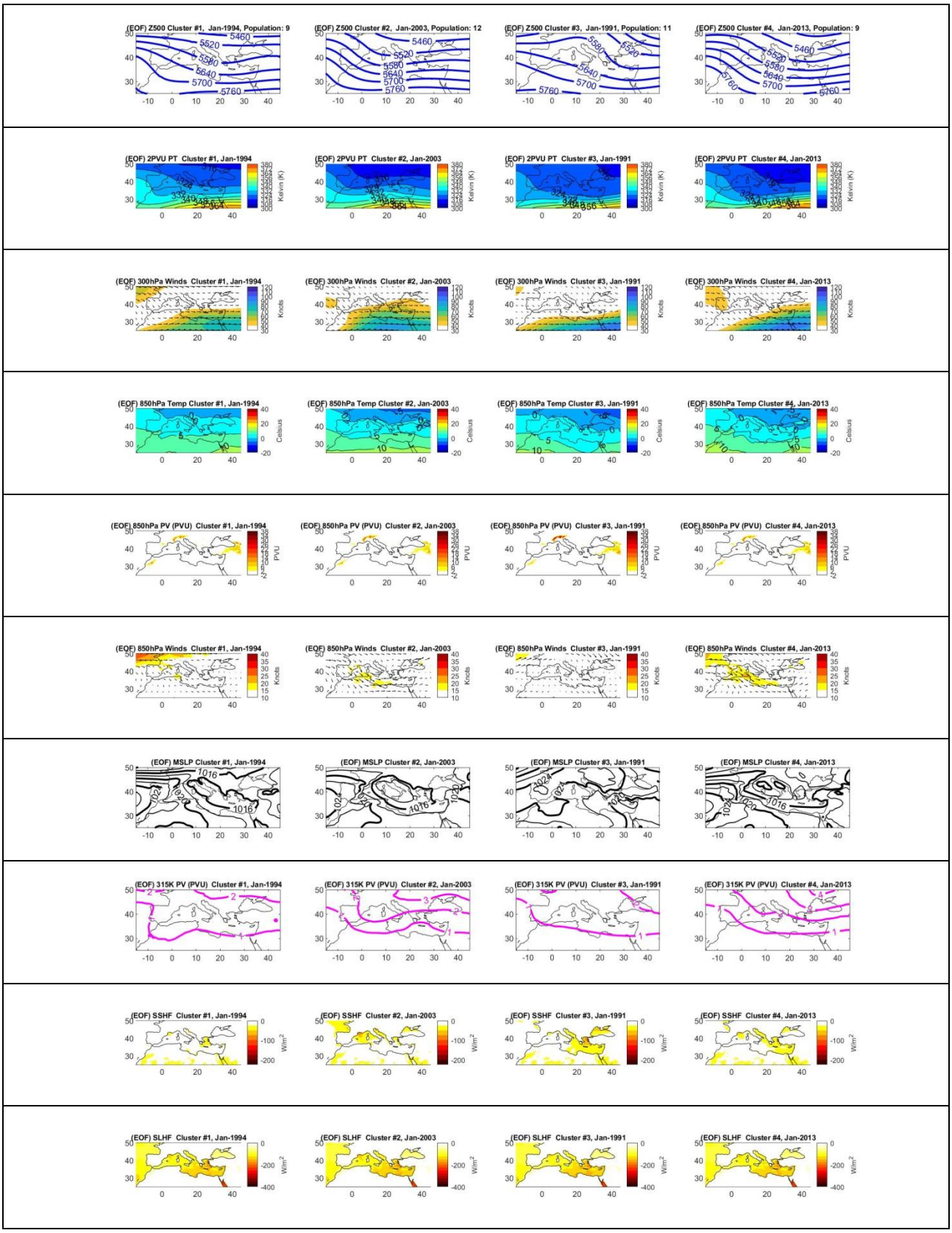


Table 16: January "weather scenarios". K-means based on Z500 observations for $\boldsymbol{k = 4}$. Cluster representatives of case "Cluster centroids of each parameter"**:**

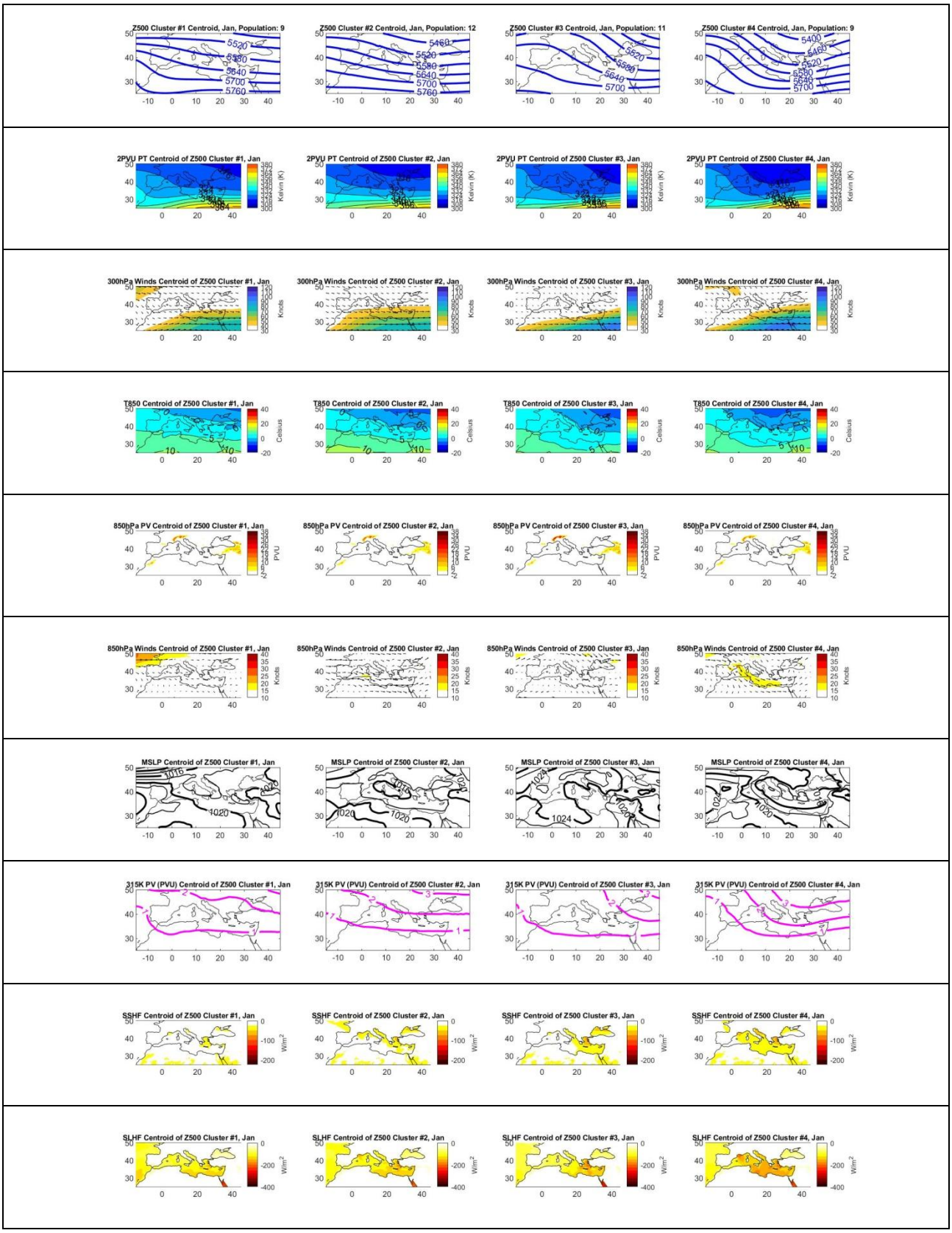


Table 17: February "weather scenarios". K-means based on Z500 observations for $\boldsymbol{k = 4}$. Cluster representatives of case "Closest to Z500 centroids (EOF)":

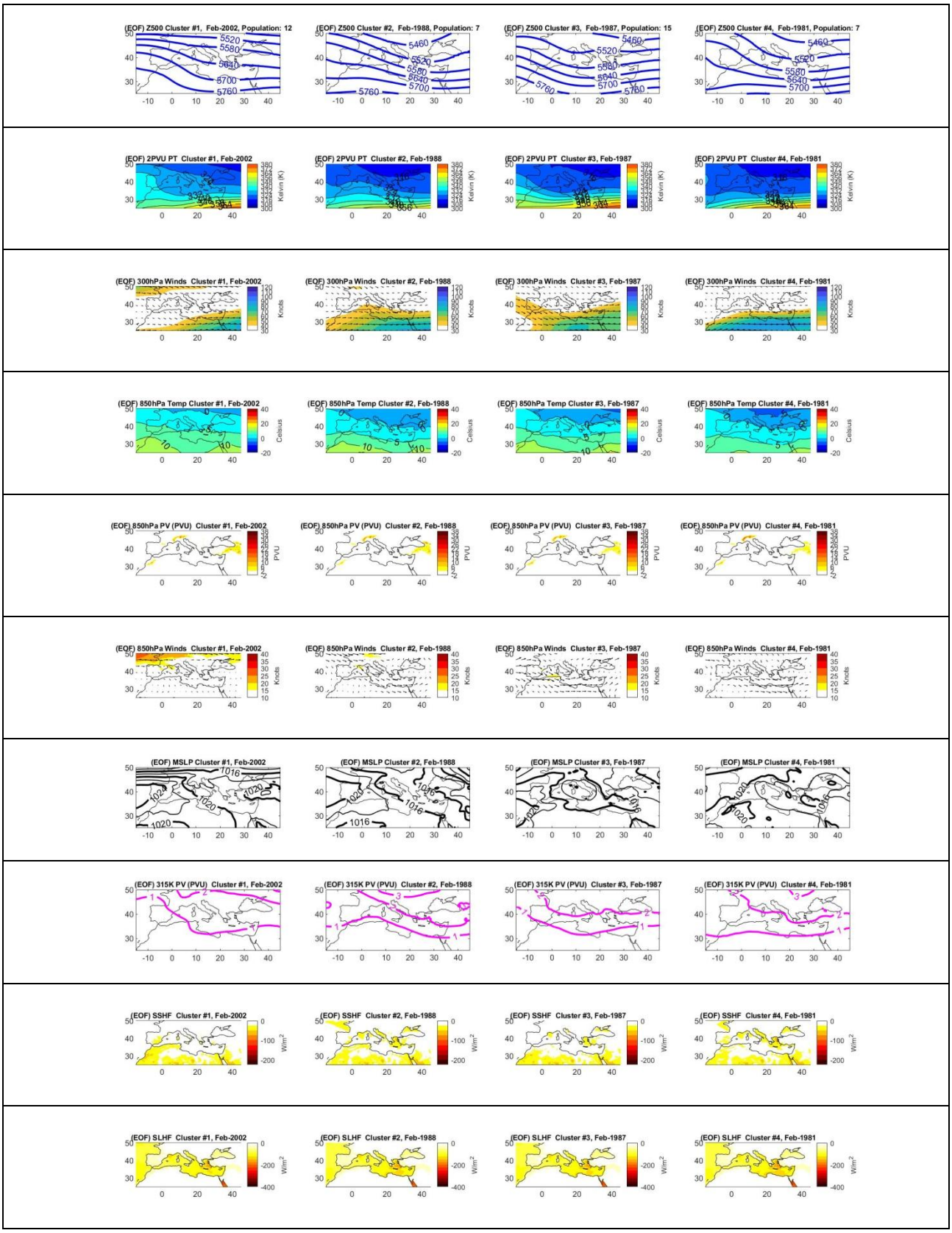


Table 18: February "weather scenarios". K-means based on Z500 observations for $\boldsymbol{k = 4}$. Cluster representatives of case "Cluster centroids of each parameter"**:**

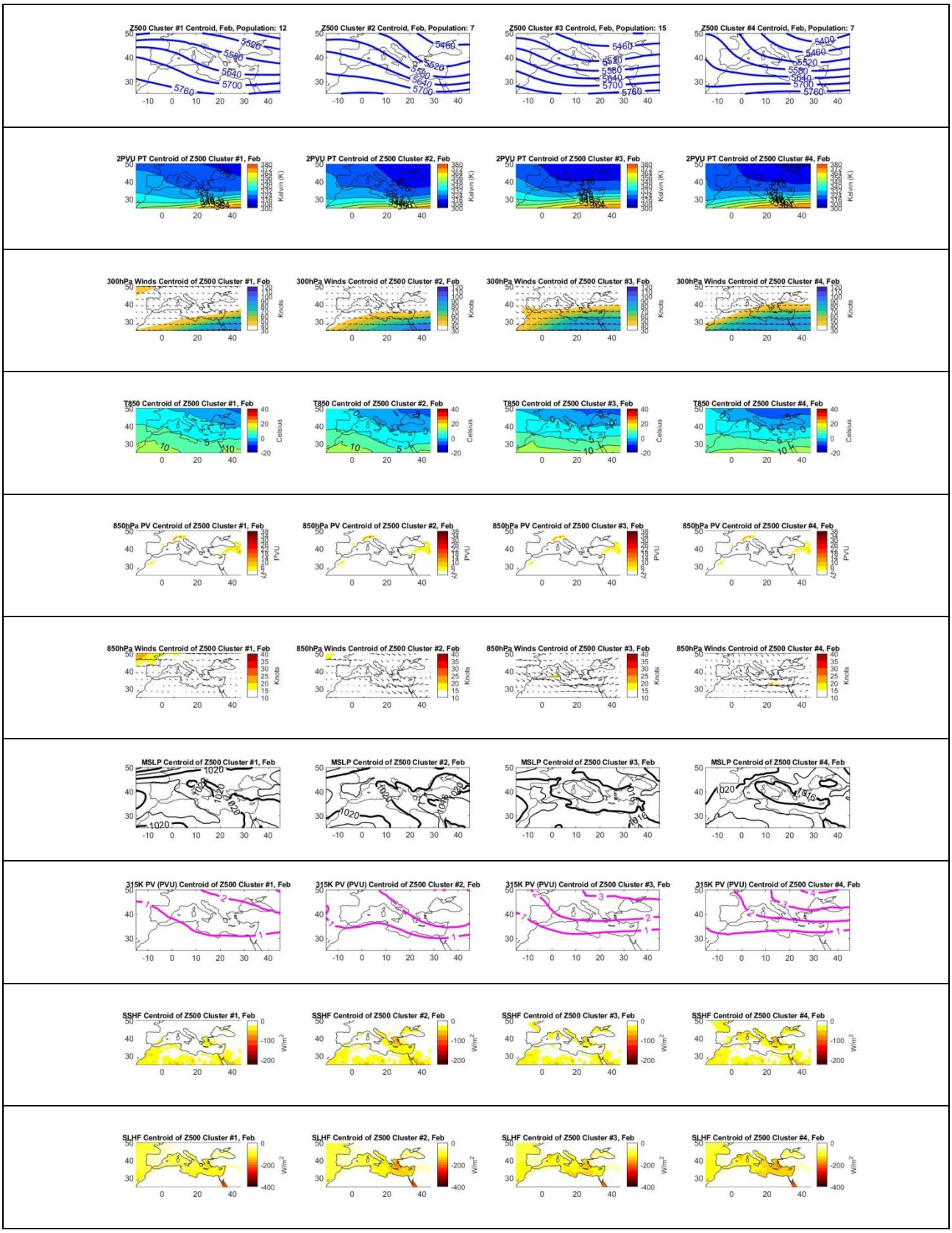


Table 19: March "weather scenarios". K-means based on Z500 observations for $k = 4$. Cluster representatives of case "Closest to Z500 centroids (EOF)":

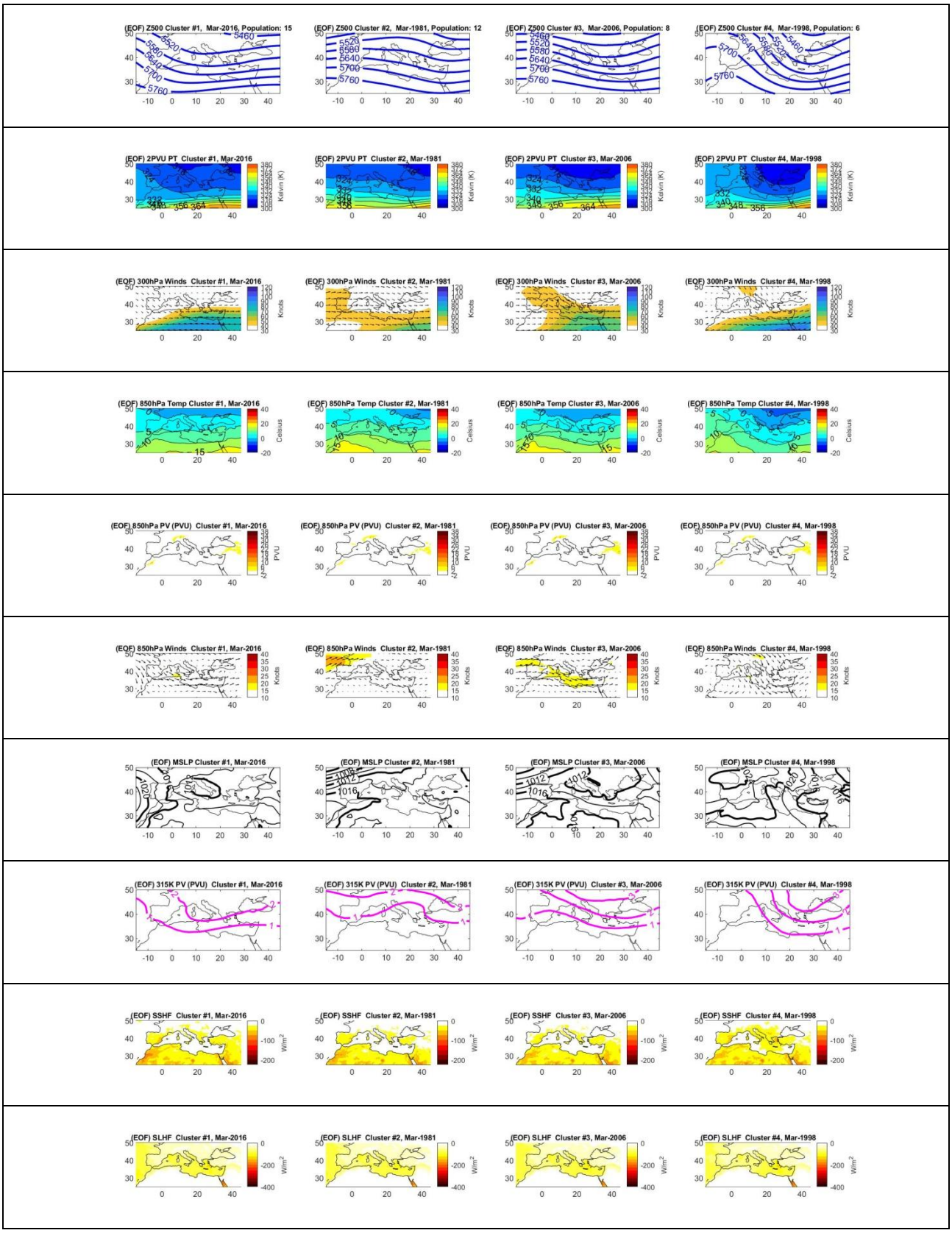
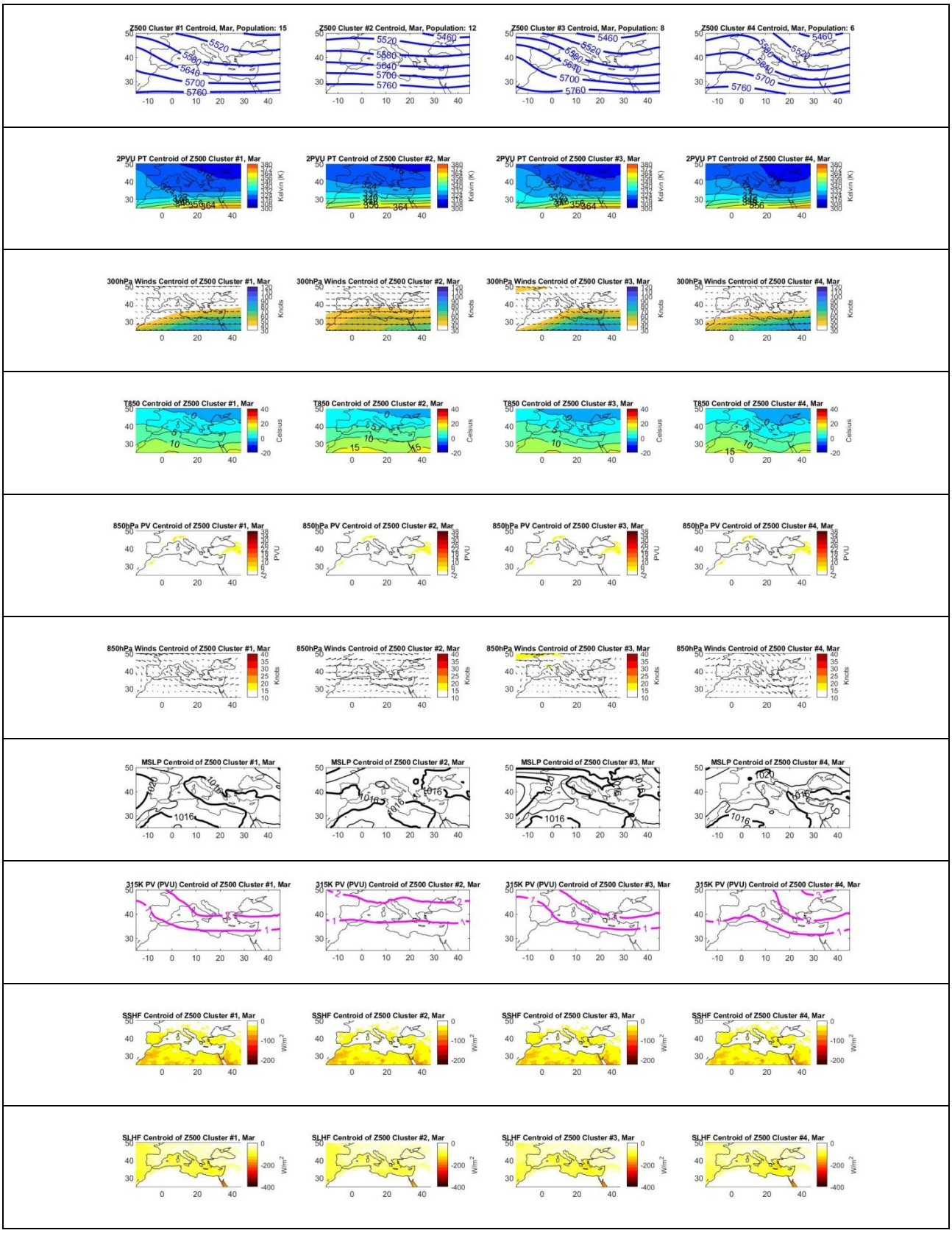


Table 20: March "weather scenarios". K-means based on Z500 observations for $\boldsymbol{k = 4}$. Cluster representatives of case "Cluster centroids of each parameter"**:**



Appendix 4: [MATLAB scripts]

Read the NETCDF files

To read the NETCDF files we use the "*ncdisp*" function. We do this for all the files of meteorological parameters.

```
filename_Z500='Z500_MODA_1979_2019.nc';

ncdisp(filename_Z500) % displays filenames properties
```

Read and store the variables of the NETCDF files

We store the variables in matrices using the "*ncread*" function. The function "*permute*" is used in order to swap the order of dimensions "*latitude*" with "*longitude*". This will help us use the "longitude" as our *x-axis* in the future. Also some calculations are made for each variable, mainly unit conversion. The variables are now matrices of dimension 101x241x492 (lat x lon x time).

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%calculate geopotential height in m, and swap dimensions lat with long
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
geopotential=double(ncread(filename_Z500,'z'));

Z500=geopotential./9.80665;
Z500=permute(Z500,[2 1 3]); %new inverted lat<->long geopotential height matrix

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%calculate MSLP in hPa, and swap dimensions lat with long
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
surf_pres=double(ncread(filename_MSLP,'msl'));

MSLP=surf_pres.*0.01;   %convert to hPa
MSLP=permute(MSLP,[2 1 3]); %new inverted lat<->long mslp matrix

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%calculate T850 Temperature in Celsius m, and swap dimensions lat with long
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Temp=double(ncread(filename_T850,'t'));

T850=Temp-273.15; %Teperature in Celsius
T850=permute(T850,[2 1 3]); %new inverted lat<->long 850hPa Temp matrix

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%calculate SLHF in W m-2, and swap dimensions lat with long
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
SLHF=double(ncread(filename_SLHF,'slhf')); % in J m-2

SLHF=SLHF./86400; % convert to W m-2 by dividing with Days=1 in seconds (W=J/s)
SLHF=permute(SLHF,[2 1 3]); %new inverted lat<->long matrix

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Calculate U, V components and Wind Magnitude in knots
%(swap dimensions lat<->long)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
U300=double(ncread(filename_W300,'u'));
V300=double(ncread(filename_W300,'v'));

U300=U300.*1.94384449;   %convert to knots
V300=V300.*1.94384449;   %convert to knots
U300=permute(U300,[2 1 3]); %new inverted lat<->long matrix
V300=permute(V300,[2 1 3]); %new inverted lat<->long matrix

W300=sqrt(U300.^2+V300.^2); %magnitude of 300hPa wind

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Calculate 850hPa Potential Vorticity (K m2 kg-1 s-1), and swap dimensions
%lat <-> long
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
PV850=double(ncread(filename_PV850_W850,'pv'));
```

```
PV850=PV850*10^6; % UNITS: 1PVU = 10^-6 K m^2 kg^-1 s^-1
PV850=permute(PV850,[2 1 3]); %new inverted lat<->long matrix

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Calculate U, V components and Wind Magnitude in knots
%(swap dimensions lat<->long)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
U850=double(ncread(filename_PV850_W850,'u'));
V850=double(ncread(filename_PV850_W850,'v'));

U850=U850.*1.94384449;   %convert to knots
V850=V850.*1.94384449;   %convert to knots
U850=permute(U850,[2 1 3]); %new inverted lat<->long matrix
V850=permute(V850,[2 1 3]); %new inverted lat<->long matrix

W850=sqrt(U850.^2+V850.^2); %magnitude of 850hPa wind

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%calculate SLHF in W m-2, and swap dimensions lat with long
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
SSHF=double(ncread(filename_SSHF,'sshf')); % in J m-2

SSHF=SSHF./86400; % convert to W m-2 by dividing with Days=1 in seconds (W=J/s)
SSHF=permute(SSHF,[2 1 3]); %new inverted lat<->long matrix

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Calculate 315K Potential Vorticity (K m2 kg-1 s-1), and swap dimensions
%lat <-> long
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
PV315=double(ncread(filename_PV315K,'pv'));

PV315=PV315*10^6; % UNITS: 1PVU = 10^-6 K m^2 kg^-1 s^-1
PV315=permute(PV315,[2 1 3]); %new inverted lat<->long matrix

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Calculate 2PVU Potential Temperature (K), and swap dimensions
%lat <-> long
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
PT2PVU=double(ncread(filename_PT2PVU,'pt'));

PT2PVU=permute(PT2PVU,[2 1 3]); %new inverted lat<->long matrix
```

The time variable

The time variable steps must be common for all the meteorological parameters. That is crucial for the SVD of coupled fields analysis. Also we need to have a common base for the variables we are referring to. From the NETCDF file info we can see that the time is measured in hours from the date 1/1/1900. So we convert to standard date format using the “datetime” and “hours” function.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%calculate time variable
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
t=double(ncread(filename_Z500,'time'))'; %time in hours as double
time=hours(t) + datetime(1900, 1, 1); %convert time+offset to datetime
```

We then split into vectors, where each vector contains the “year”, “month”, “day”, “hours” of the observation. This is done in order to split the observations in sets of the same month. This will help us calculate the monthly means over a period of 41 years.

We convert the variables to “double” again. The variable “*t*” will be used for the monthly means. The variable “*t2*” is the time variable containing the time steps of observations of the various meteorological parameters used.

```
[year,month,day,hours]=datevec(time); %split datetime into year, month, day, hours vectors
[year2,month2,day2,hours2]=datevec(time); %split datetime into year, month, day, hours vectors
%change all the years to be the same in order to take the monthly mean
%throughout all the given years
```

```
count=0;
for iii=1:(max(year)-min(year)+1)
    year(year==min(year)+count)=min(year);
    count=count+1;
end

t=double(datenum(datetime(year, month, day))); % convert to double again...
                                               %with all years changed to min(year)
t2=double(datenum(datetime(year2, month2, day2))); % convert to double...
                                                   %again of original time series
```

Read and store the latitude and longitude variables

Read, store and display in the command line the latitude and longitude variables.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%calculate lat and long variables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
longitude=double(ncread(filename_Z500,'longitude'));
latitude=double(ncread(filename_Z500,'latitude'));

disp(['min longitude=',num2str(min(longitude)),'   ','max
longitude=',num2str(max(longitude))]);
disp(['min latitude=',num2str(min(latitude)),'    ','max latitude=',num2str(max(latitude))]);
```

The Downsample function

From this monthly dataset, this function[10] is used to calculate the average at each grid cell for any given month. Returns the monthly means from a 41 year period (101x241x12).

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%use of function downsample for statistical basics
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[Z500_years_monthlymean,t_monthly] = downsample_ts(Z500,t,'month','mean');
```

The same applies for the rest of the meteorological parameters.

Calculate the fields of anomalies

If we remove the seasonal cycle (the average at each grid cell for any given month) of each meteorological variable from the variable itself, we will get the fields of anomalies (the deviation from the mean).

This is done with the function “Anomalies”

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%calculate anomalies per month compared to all years monthly mean
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Z500_anomalies=Anomalies(longitude,latitude,t2,Z500,Z500_years_monthlymean);
```

Below is the function.

```
function anomalies = Anomalies(X,Y,T,monthly_field,years_monthly_mean)
%X: is the longitude vector
%Y: is the latitude vector
%T: is the time vector (Serial Date Number format)
%anomalies: returns the anomalus field matrix

time=datetime(T,'ConvertFrom','datenum'); %convert T vector from Serial Date Number to
Datetime format
[year,month,day]=datevec(time); %split datetime into year, month, day, hours vectors
```

[10] Chad Greene (2021). downsample_ts, MATLAB Central File Exchange. Retrieved January 18, 2021. https://www.mathworks.com/matlabcentral/fileexchange/48361-downsample_ts

```
anomalies=zeros(length(Y),length(X),length(T));

%loop for all the distinct months, for each year and subtract the years_monthlymean
count=1;
for j=min(year):max(year)
    for k=min(month):max(month)
        for l=min(day):max(day)
            index= T==double(datenum(datetime(j,k,l))); %logical indexing of array t

            A=squeeze(anomalies(:,:,index));
            B=squeeze(monthly_field(:,:,index));
            C=squeeze(years_monthly_mean(:,:,k));
            A=B-C;

            for ii=1:length(Y)
               for jj=1:length(X)
                  anomalies(ii,jj,count)=A(ii,jj);
               end
            end
            count=count+1;
        end

    end
end

end
```

This is repeated for all the meteorological parameters.

EOF analysis

In order to perform EOF analysis (section 1.2.1) of the fields of anomalies, we call the "eof" function[11].

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% EOF Analysis
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
N_EOFs=4; %number of EOFs and PCs modes, to plot

[eof_maps_Z500,pc_Z500,expv_Z500] = eof(Z500_anomalies); %needs the Climate Data Toolbox
[eof_maps_norm_Z500,pc_norm_Z500] =
EOF(Z500,longitude,latitude,t2,eof_maps_Z500,pc_Z500,expv_Z500,N_EOFs);
```

The "EOF" function shown bellow is to normalize the EOFs and PCs (section 1.2.3), and also to plot the leading "N_EOFs" modes.

```
function [eof_maps_norm,pc_norm] = EOF(param,lon,lat,T,eof_maps,pc,expv,Modes)
%UNTITLED2 Summary of this function goes here
%param: the parameter to perform EOF analysis (lat,long,t)
%T: the time variable array of 'param'
%lon: longitude array (common for both X and Y)
%lat: latitude array (common for both X and Y)
%eof_maps : the EOFs of the field of anomalies of the parameter
%pc : the PCs of the field of anomalies of the parameter
%expv: the fractions of the total variability, of each mode of the EOFs
%Modes: the first EOF modes to plot
%pc_norm : scale pc range: [-1 1]
%eof_maps_norm : adjusted EOFs according to the 'pc_norm'

    a=inputname(1); %string with the parameters name

    %scale pc range: [-1 1]
    pc_norm=zeros(size(pc));
    eof_maps_norm=zeros(size(eof_maps));
    for k = 1:size(pc,1)
        % Find the the maximum value in the time series of each principal component:
        [maxval,ind] = max(abs(pc(k,:)));
```

[11] Greene, C. A., Thirumalai, K., Kearney, K. A., Delgado, J. M., Schwanghart, W., Wolfenbarger, N. S., et al. (2019). The Climate Data Toolbox for MATLAB. Geochemistry, Geophysics, Geosystems, 20. https://doi.org/10.1029/2019GC008392

```
        % Divide the time series by its maximum value:
        pc_norm(k,:) = pc(k,:)/maxval;

        % Multiply the corresponding EOF map:
        eof_maps_norm(:,:,k) = eof_maps(:,:,k)*maxval;
    end

    %scale EOF maps to range: [-1 1]
    eof_maps_plot=zeros(size(eof_maps));
    for k=1:size(pc,1)
        [maxval,ind] = max(abs(eof_maps(:,:,k)));
        eof_maps_plot(:,:,k)=eof_maps(:,:,k)/max(maxval);
    end

 if (Modes>0)
 %% Plot the Leading N PCs and EOFs, Normalized to range [-1 1]
    figure('Name','PCs and EOFs Normalized to range [-1 1]','NumberTitle','off');
    Coast = load('coast.mat');
    count=1;
    for k=1:2:2*Modes

%%%%%%% Plot the EOFs scaled to range [-1 1] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        subplot(Modes,2,k)

        %pcolor
        number_of_shades=5; % number of color shades
        blue=linspace(0.3,1,number_of_shades); %Increase the first value for lighter blue at
end,
                                               %increase the second value for lighter blue at
the beginning
        red =linspace(1,0.3,number_of_shades); %Increase the first value for lighter red at
beginning,
                                               %increase the second value for lighter red at
the end
        neg_map=[blue',blue',ones(number_of_shades,1)]; %shades of blue color map matrix
        pos_map=[ones(number_of_shades,1),red',red']; %shades of red color map matrix
        colormap([neg_map;pos_map]) %apply the above colormaps

        pcolor(lon, lat, eof_maps_plot(:,:,count)); shading interp;
        minc=-1;
        maxc= 1;
        caxis([minc maxc]) % pcolor data range
        hold on

        %Contrours
        clow=minc; czero=0; chigh=maxc; cint=0.2;     % contour line limits
        conts1 = [clow:cint:czero-cint];            % define contour lines negative
        conts2 = [czero+cint:cint:chigh];           % define contour lines positive

        [a1,b1]=contour(lon,lat,eof_maps_plot(:,:,count),conts1,'blue'); %construct the
negative isolines
        b1.LineWidth = 0.5;
        clabel(a1,b1,'Color','blue')                  %Label negative contour plot elevation

        [a2,b2]=contour(lon,lat,eof_maps_plot(:,:,count),conts2,'red');  %construct the
positive isolines
        b2.LineWidth = 0.5;
        clabel(a2,b2,'Color','red')                   %Label positive contour plot elevation
        hold on

        %Coastlines
        plot(Coast.long,Coast.lat,'k')
        axis equal
        xlim([min(lon) max(lon)])
        ylim([min(lat) max(lat)])
        hold on

        title([a,' EOF #',num2str(count),' ',' (',num2str(expv(count),'%0.1f'),'%)']);

%%%%%%% Plot the PCs scaled to range [-1 1] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        subplot(Modes,2,k+1)

        plot(T,movmean(pc_norm(count,:),13)) %plot the 13-month moving mean (smoother)
        datetick('x','yyyy','keeplimits')
        title([a,' PC #',num2str(count),' ',' (',num2str(expv(count),'%0.1f'),'%)']);
        legend(a,'Location','northwest')
        ylim([-0.5 0.5])
```

```
        count=count+1;
    end
 end
end
```

PCs in order to span the range [-1, 1] are scaled. The matrix containing the scaled PCs is "`pc_norm`". The corresponding EOFs are adjusted accordingly ("`eof_maps_norm`").

For the desired limit of variability described from the EOF modes, the script below returns the number modes needed.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Number of EOF Modes corresponding to (limit)% of the variability of
% the original data (returns eof_modes)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
limit = 90; %min desired limit of variability percentage (EOFs)

total_percentage_Z500=0;
for k = 1:size(pc_Z500,1)
    total_percentage_Z500=total_percentage_Z500+expv_Z500(k);
    eof_modes=k;
if total_percentage_Z500 > limit
        break
    end
end

disp(' ');
disp(['The first ',num2str(eof_modes),' EOF modes correspond to :',]);
disp([num2str(total_percentage_Z500,'%0.3f'),'%',' ','of the Variability of the original Z500
data']);
```

This is repeated for all the meteorological parameters.

SVD of coupled fields analysis

For a given pair of meteorological variables, based on section 1.2.2, the function "SVD" is used in order to:

- Calculate the Square-covariance fraction explained by each singular mode
- Calculate the SVD maps of each variable
- Calculate the SVD PCs of each variable
- Calculate the Pearson's correlation coefficient between the PCs of two variables (for the same mode of square-covariability)
- Calculate the P-values
- Scale or Standardize PCs in order to plot them for comparison
- Scale the SVD maps in order to plot them for comparison
- Plot the first *N* modes of the variables in pairs.

In order for the build-in "svd" function to be faster, we make use cubic interpolation in order to reduce the grid points (resolution and data volume).

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SVD Analysis X Vs Y (anomalous fields) with common T (time dimension)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function SVD(nameX,nameY,X_anom,Y_anom,T,lon,lat,Modes,InterPoints,Standardize)
%Modes: the first SVD modes to plot
%InterPoints: new points x and y coordinates to interpolate (reduce matrix size)
%nameX: the first parameters name
%nameY: the second parameters name
%X_anom: the field of anomalies of the first parameter
%Y_anom: the field of anomalies of the second parameter
%T: the time variable array (common time steps for X and Y)
```

```
%lon: longitude array (common for both X and Y)
%lat: latitude array (common for both X and Y)
%Standardize exp. coefs(3 options):
%           ==1: z-score standardization (by mean and st. deviation)
%           ==2: scaled in the range of [-1 1]
%           ==0: no standardization (Raw)

    svd_modes=Modes; %Number of modes to plot
    newpoints = InterPoints; % points to interpolate for reduction of resolution (faster)
    S=Standardize;
    a=nameX;
    b=nameY;

    %Interpolate data to reduce size of X_anom & Y_anom matrices
    %newpoints = 50;
    x=zeros(newpoints,newpoints,length(T));
    y=zeros(newpoints,newpoints,length(T));
    for i=1:length(T)
        [xq,yq] = meshgrid(...
                linspace(min(min(lon,[],2)),max(max(lon,[],2)),newpoints ),...
                linspace(min(min(lat,[],1)),max(max(lat,[],1)),newpoints )...
              );
        x(:,:,i) = interp2(lon,lat,X_anom(:,:,i),xq,yq,'cubic');
        y(:,:,i) = interp2(lon,lat,Y_anom(:,:,i),xq,yq,'cubic');
    end

    %Reshape to 2D & Permute (nxp)
    x = permute(x,[3 1 2]);
    x = reshape(x,size(x,1),size(x,2)*size(x,3));%2D X anomalies matrix
    y = permute(y,[3 1 2]);
    y = reshape(y,size(y,1),size(y,2)*size(y,3));%2D Y anomalies matrix

    x=detrend(x,0);% Remove the time mean of each column
    y=detrend(y,0);% Remove the time mean of each column

    SC=x'*y; %Cross-Covariance Matrix or Square Cov. matrix (SC)
    tic;

    %[U,L,V]=svd(SC);        %SVD equal to : C=U*L*V'
                            %columns of U: singular vectors of x (spatial pattern - standing
oscilation)
                            %columns of V: singular vectors of y (spatial pattern - standing
oscilation)
    [U,L,V]=svd(SC,'econ');%The economy-size SVD decomposition of 'SC' removes...
                            %extra rows or columns of zeros from the diagonal...
                            %matrix of singular values, S, along with the ...
                            %columns in either U or V that multiply those zeros...
                            %in the expression C = U*L*V'.
    toc
    A = x*U; %columns of A: expansion coefs of x (time series)
    B = y*V; %columns of B: expansion coefs of y (time series)

    l=diag(L);
    SCF=100*((l.^2)./sum(l.^2))'; %percentage of Squared Variance explained by...
                                  %the corresponding singular vectors

%%% Calculate the SVD maps (spatial pattern - standing oscillation) %%%%%%%
    SVD_maps_S=U'; %lines are the singular vectors
    SVD_maps_P=V'; %lines are the singular vectors

    SVD_maps_S=reshape(SVD_maps_S,size(SVD_maps_S,1),length(xq),length(yq)); %Un-reshape
    SVD_maps_P=reshape(SVD_maps_P,size(SVD_maps_P,1),length(xq),length(yq)); %Un-reshape

    SVD_maps_S=permute(SVD_maps_S,[2 3 1]); %Un-permute
    SVD_maps_P=permute(SVD_maps_P,[2 3 1]); %Un-permute

    %SVD_maps_S(:,:,i): is the SVD_map of variable 'S' corresponding to mode 'i'


%%% Calculate Total Cross Covariance Matrix Percentage described %%%%%%%%%%
    SC_percentage=0;
    for k = 1:svd_modes
        SC_percentage=SC_percentage+SCF(k);
    end
    disp(['Total Percentage of Cross Covariance Matrix described by
SVD=',num2str(SC_percentage)]);
```

```
if (svd_modes>0)
%%% Plot the leading Expansion Coefficients modes %%%%%%%%%%%%%%%%%%%%%%%%%%
    figure('Name','SVD PCs','NumberTitle','off');
    for k=1:svd_modes
        if S==1
            %SVD expansion coeficients Normalized by Mean and Standard Deviation (z-score)
            A_norm=zscore(A(:,k)); %Z-score normalization z=(X-μ)/σ
            B_norm=zscore(B(:,k)); %Z-score normalization z=(X-μ)/σ
            text='Standardized';
        elseif S==2
            %SVD expansion coefficients Scaled to range [-1 1]
            A_norm = A(:,k)/max(abs(A(:,k)));
            B_norm = B(:,k)/max(abs(B(:,k)));
            text='Scaled';
        else
            %SVD expansion coefficients RAW
            A_norm = A(:,k);
            B_norm = B(:,k);
            text='Raw';
        end

        %Pearson's correlation coef and P-value calculation
        %if 'P-value' is less than ex. 0.05 (95% significance level),
        %then the correlation 'r' is significantly different from zero.
        %meaning that There IS A SIGNIFICANT LINEAR RELATIONSHIP (correlation)
        %between x and y VECTORS
        [r,pval]=corr(A_norm,B_norm,'Type','Pearson');

        subplot(svd_modes,1,k)

        plot(T,movmean(A_norm,13)) %plot the 13-month moving mean (smoother)
        hold on
        plot(T,movmean(B_norm,13)) %plot the 13-month moving mean (smoother)
        hold off
        datetick('x','yyyy','keeplimits')
        title([text,' ','Exp Coefs #',num2str(k),' ',' (',num2str(SCF(k),'%0.1f'),'%)',' ',...
            'r=',num2str(r,'%.2f'),'(',num2str(pval),')']);
        legend(a,b,'Location','northwest')
        if S==1
            ylim([-1 1])
        elseif S==2
            ylim([-0.5 0.5])
        else
            %nothing
        end

    end

%%% Plot SVD maps of the leading modes %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    figure('Name','SVD maps Scaled to range [-1 1]','NumberTitle','off');
    Coast = load('coast.mat');
    count=1;
    for k=1:2:2*svd_modes

        %scale SVD maps to range: [-1 1]
        [maxval_S,ind_S] = max(abs(SVD_maps_S(:,:,count))); %maxval_S: max values of columns
of 'SVD_maps_S'
        [maxval_P,ind_P] = max(abs(SVD_maps_P(:,:,count))); %maxval_P: max values of columns
of 'SVD_maps_P'
        SVD_maps_S_norm=SVD_maps_S(:,:,count)/max(maxval_S); %divide 'SVD_maps_S' by the max
of maxval_S
        SVD_maps_P_norm=SVD_maps_P(:,:,count)/max(maxval_P); %divide 'SVD_maps_P' by the max
of maxval_P

%%%%%%% Plot the 'SVD_maps_S' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        subplot(svd_modes,2,k) % first column of subplot
        %Colormap
        number_of_shades=5; % number of color shades
        blue=linspace(0.3,1,number_of_shades); %Increase the first value for lighter blue at
end,
                                               %increase the second value for lighter blue at
the beginning
        red =linspace(1,0.3,number_of_shades); %Increase the first value for lighter red at
beginning,
                                               %increase the second value for lighter red at
the end
        neg_map=[blue',blue',ones(number_of_shades,1)]; %shades of blue color map matrix
```

```
        pos_map=[ones(number_of_shades,1),red',red']; %shades of red color map matrix
        colormap([neg_map;pos_map]) %apply the above colormaps

        %P-color
        pcolor(xq, yq, SVD_maps_S_norm); shading interp;
        minc=-1;
        maxc= 1;
        caxis([minc maxc]) % pcolor data range
        hold on

        %Contrours
        clow=minc; czero=0; chigh=maxc; cint=0.2;     % contour line limits
        conts1 = [clow:cint:czero-cint];            % define contour lines negative
        conts2 = [czero+cint:cint:chigh];           % define contour lines positive
        [a1,b1]=contour(xq,yq,SVD_maps_S_norm,conts1,'blue'); %construct the negative isolines
        b1.LineWidth = 0.5;
        clabel(a1,b1,'Color','blue')                  %Label negative contour plot elevation
        [a2,b2]=contour(xq,yq,SVD_maps_S_norm,conts2,'red');  %construct the positive isolines
        b2.LineWidth = 0.5;
        clabel(a2,b2,'Color','red')                   %Label positive contour plot elevation
        hold on

        %Coastlines
        plot(Coast.long,Coast.lat,'k')
        axis equal
        xlim([min(lon) max(lon)])
        ylim([min(lat) max(lat)])
        hold on

        %Title
        title([a,' SVD map #',num2str(count),' ','(',num2str(SCF(count),'%0.1f'),'%)']);

%%%%%%% Plot the 'SVD_maps_P' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        subplot(svd_modes,2,k+1) % second column of subplot

        %pcolor
        pcolor(xq,yq, SVD_maps_P_norm); shading interp;
        minc=-1;
        maxc= 1;
        caxis([minc maxc]) % pcolor data range
        hold on

        %Contrours
        clow=minc; czero=0; chigh=maxc; cint=0.2;     % contour line limits
        conts1 = [clow:cint:czero-cint];            % define contour lines negative
        conts2 = [czero+cint:cint:chigh];           % define contour lines positive

        [a1,b1]=contour(xq,yq,SVD_maps_P_norm,conts1,'blue'); %construct the negative isolines
        b1.LineWidth = 0.5;
        clabel(a1,b1,'Color','blue')                  %Label negative contour plot elevation

        [a2,b2]=contour(xq,yq,SVD_maps_P_norm,conts2,'red');  %construct the positive isolines
        b2.LineWidth = 0.5;
        clabel(a2,b2,'Color','red')                   %Label positive contour plot elevation
        hold on

        %Coastlines
        plot(Coast.long,Coast.lat,'k')
        axis equal
        xlim([min(lon) max(lon)])
        ylim([min(lat) max(lat)])
        hold on

        %Title
        title([b,' SVD map #',num2str(count),' ','(',num2str(SCF(count),'%0.1f'),'%)']);

        count=count+1;
    end
end
end
```

K-means clustering

Based on section 1.2.4, using MATLAB's "Statistics and Machine Learning Toolbox"[12]. Instead of this part of code, we can use the script for SOM (Appendix 5: [MATLAB script for SOM]). The rest are common for both algorithms (k-means or SOM).

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% K-means monthly Clustering of Z500 EOF data & of Z500 Full Patterns
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
k=4;                        %Number of Clusters to divide data
Replicates=1000; % Number of times to repeat K-means algorithm
%CRM=2; %cluster representatives mode:
      %1. plot the full fields closest to centroids
      %2. plot the full fields (constructing them from the EOFs and the...
                            %seasonal mean) closest to centroids (smoother)
      %3. plot the full parameter fields at Z500 cluster centroids
      %4. find the parameter PCs belonging in the same Z500 cluster,
      %find its centroid, plot at those centroids (centroids within Z500 cluster)

orientation=0; %How to plot the parameters:
               %landscape==0 / portrait== anything else

ci=zeros(length(t2),1);     %Vector with cluster indices

% loop for each month %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for m=min(month2):max(month2)
    close all

    %if i want to split all EOF (ex Januaries) into k clusters:
    index= find(month2==m);        %January position index in t2 array
    pc_month=pc_Z500_norm(1:eof_modes,index)'; %Principle Components containing only...
                                   %the Januaries, Transpose (: columns are...
                                   %the variables and rows are the points)
    %
    [idx,C,sumd,D] = kmeans(pc_month,k,'Replicates',Replicates);
      % split all Januaries into k Clusters and so on for other months
      % idx: vector with cluster indices that each time step belongs
      % C  : matrix of which rows are cluster centroid coordinates...
      %                                  (in a "eof_modes" Dimensional space)
      %sumd: returns the within-cluster sums of point-to-centroid distances in the k-by-1
vector
      %D: returns distances from each point to every centroid in the n-by-k matrix
    %
    disp(' ');
    disp(['The Best total sum of K-mean distances = ',num2str(sum(sumd))]);

%%% Find the population number of each cluster %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    population=zeros(1,k);
    for i=1:k
        population(i)=length(find(idx==i));
    end
    population;

%%% Cluster Reference Full Pattern %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %Find the full pattern time step closest to the centroids.

    [M,I]=min(D);    %M:the min of each column in cluster i
                     %I:index of the first occurrence of min (in case...
                     %  multiple distances of columns of D (same cluster)...
                     %  are equal, keep only the fist occurrence and its index
    %so the most representative full Z500 patterns are for example:
                         %- the 15th January from the 41(years) in total
                         %- the 39th January from the 41 in total
                         %- the 16th January from the 41 in total
                         %- the  1st January from the 41 in total
    %each belonging to deferent cluster

    crti=index(I);       % cluster reference time indices in t2 array
    ci(index)=idx;       % update the cluster indices (ci) vector
    datestr(t2(crti))    % to see the dates of min Distance from centroid
end
```

[12] https://www.mathworks.com/products/statistics.html
https://www.mathworks.com/help/stats/kmeans.html

The cluster representatives

The scripts "PLOT_CRM1.m", "PLOT_CRM2.m", "PLOT_CRM3.m", "PLOT_CRM4.m" are for calculating the cluster representatives of each parameter and for plotting them. There are four options available (section 1.2.6):

```
%%%% Plot the cluster representatives full field (4 options to plot) %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    if CRM==1      %mode 1:
                   %plot the full fields closest to centroids. Here
                   %we use the closest Z500 pattern to the cluster
                   %centroids. And for the same date, we plot mslp and T850.

        open PLOT_CRM1.m
        PLOT_CRM1

    elseif CRM==2 %mode 2:
                   %plot the full fields (constructing them from...
                   %the EOFs and the seasonal mean) closest to centroids (smoother)

        open PLOT_CRM2.m
        PLOT_CRM2

    elseif %mode 3:
         %-plot the full Z500 cluster centroids %%%%%%%%%%%%%%%%%%%%
         %here we will try to use the Z500 cluster centroids for projecting
         %mslp and T850 on them. The results are promising due to high
         %correlation between PCs from Z500-T850-MSLP-SLHF. DO not forget to
         %normalize all PCs before using Cluster centroids for other than
         %Z500.
%

        open PLOT_CRM3.m
        PLOT_CRM3

    else %mode 4:
         %-plot the Z500 full field using Z500 cluster centroids
         %-plot the other parameters full fields using the centroids of
         %          parameters PCs that belong in the same Z500 cluster
         %-from Kmeans_centroid_Vs_representative_test.m we see that for
         %eof_modes=20 the parameters centroids(as described above) have
         %the min distance from Z500 cluster centroids (ex for eof_modes=5
         %the Z500 representatives are better choice)

        open PLOT_CRM4.m
        PLOT_CRM4

    end
```

The "closest to Z500 centroids" cluster representatives

The script "PLOT_CRM1.m" calculates and plots the cluster representatives of this case:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% K-means monthly Clustering of Z500 EOF data & of Z500 Full Patterns
CRM=1;  %cluster representatives mode:
      %1. plot the full fields closest to centroids
      %2. plot the full fields (constructing them from the EOFs and the...
                          %seasonal mean) closest to centroids (smoother)
      %3. plot the full parameter fields at Z500 cluster centroids
      %4. find the parameter PCs belonging in the same Z500 cluster,
      %find its centroid, plot at those centroids (centroids within Z500 cluster)

%orientation  ==0 :   landscape
%             ==1,2,3,... : default (portrait)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%close all
%Coast = load('coast.mat');

figure('Name',['Z500 Cluster Representatives for the month',...
```

```
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PLOT_Z500(longitude,latitude,Z500(:,:,i))

    %plot title%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %DateString = datestr(t_monthly(i),'dd-mmm-yyyy HH:MM:SS');
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['Z500 Cluster #',num2str(ci(i)),'  ', DateString,', ','Population:
',num2str(population(ci(i)))],'FontSize',8);

    count=count+1;
end
f1=gcf;

figure('Name',['PT2PVU at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PLOT_PT2PVU(longitude,latitude,PT2PVU(:,:,i),5)

    %title
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['2PVU PT','  ','Cluster #',num2str(ci(i)),' ', DateString]);

    count=count+1;
end
f2=gcf;

figure('Name',['W300 at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PLOT_W300(longitude,latitude,U300,V300,U300(:,:,i),V300(:,:,i),W300(:,:,i),1)

    %title
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['300hPa Winds','  ','Cluster #',num2str(ci(i)),' ', DateString]);
    count=count+1;
end
f3=gcf;

figure('Name',['T850 at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PLOT_T850(longitude,latitude,T850(:,:,i))

    %plot title %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['850hPa Temp Cluster #',num2str(ci(i)),'  ', DateString]);

    count=count+1;
```

```
end
f4=gcf;

figure('Name',['PV850 at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PLOT_PV850(longitude,latitude,PV850(:,:,i))

    %title
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['850hPa PV (PVU)','  ','Cluster #',num2str(ci(i)),' ', DateString]);

    count=count+1;
end
f5=gcf;

figure('Name',['W850 at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PLOT_W850(longitude,latitude,U850,V850,U850(:,:,i),V850(:,:,i),W850(:,:,i),1)

    %title
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['850hPa Winds','  ','Cluster #',num2str(ci(i)),' ', DateString]);

    count=count+1;
end
f6=gcf;

figure('Name',['MSLP at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PLOT_MSLP(longitude,latitude,MSLP(:,:,i),5)

    %title
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['MSLP Cluster #',num2str(ci(i)),'  ', DateString]);

    count=count+1;
end
f7=gcf;

figure('Name',['PV315 at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PLOT_PV315(longitude,latitude,PV315(:,:,i),5)

    %title
    DateString = datestr(t2(i),'mmm-yyyy');
```

```
    title(['315K PV (PVU)','  ','Cluster #',num2str(ci(i)),' ', DateString]);

    count=count+1;
end
f8=gcf;

        figure('Name',['SSHF at the same date as Z500 Cluster Representatives for the
month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PLOT_SSHF(longitude, latitude, SSHF(:,:,i),2)

    %Title
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['SSHF','  ','Cluster #',num2str(ci(i)),' ', DateString]);

    count=count+1;
end
f9=gcf;

figure('Name',['SLHF at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PLOT_SLHF(longitude, latitude,SLHF(:,:,i),2)

%%% Title %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['SLHF','  ','Cluster #',num2str(ci(i)),' ', DateString]);
    %xlabel('longitude (degrees North)'), ylabel('latitude (degrees East)')
    hold off
    count=count+1;
end
f10=gcf;


%%% Save every figure as jpg picture %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if orientation==0

    set(f1, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]); %1/3 landscape screen
    set(f2, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]); %2/3 landscape screen
    set(f3, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]); %3/3 landscape screen

    set(f4, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]);
    set(f5, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]);
    set(f6, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]);

    set(f7, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]);
    set(f8, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]);
    set(f9, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]);

    set(f10, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]); %3/3 landscape screen

else
    set(f1, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f2, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f3, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f4, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f5, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f6, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f7, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
```

```
    set(f8, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f9, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f10, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %1/3 portrait screen
end

%SAVE FIGURES TO FILE
%
savename_string1 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_closest to Z500
centroid_Z500','.jpg'];
savename_string2 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_closest to Z500
centroid__PT2PVU','.jpg'];
savename_string3 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_closest to Z500
centroid__W300','.jpg'];
savename_string4 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_closest to Z500
centroid__T850','.jpg'];
savename_string5 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_closest to Z500
centroid__PV850','.jpg'];
savename_string6 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_closest to Z500
centroid__W850','.jpg'];
savename_string7 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_closest to Z500
centroid__MSLP','.jpg'];
savename_string8 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_closest to Z500
centroid__PV315','.jpg'];
savename_string9 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_closest to Z500
centroid__SSHF','.jpg'];
savename_string10 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_closest to Z500
centroid__SLHF','.jpg'];
print(f1,savename_string1, '-djpeg')
print(f2,savename_string2, '-djpeg')
print(f3,savename_string3, '-djpeg')
print(f4,savename_string4, '-djpeg')
print(f5,savename_string5, '-djpeg')
print(f6,savename_string6, '-djpeg')
print(f7,savename_string7, '-djpeg')
print(f8,savename_string8, '-djpeg')
print(f9,savename_string9, '-djpeg')
print(f10,savename_string10, '-djpeg')
%
```

The "closest to Z500 centroids (EOF)" cluster representatives

The script "`PLOT_CRM2.m`" calculates and plots the cluster representatives of this case:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% K-means monthly Clustering of Z500 EOF data & of Z500 Full Patterns
CRM=2;  %cluster representatives mode:
      %1. plot the full fields closest to centroids
      %2. plot the full fields (constructing them from the EOFs and the...
                            %seasonal mean) closest to centroids (smoother)
      %3. plot the full parameter fields at Z500 cluster centroids
      %4. find the parameter PCs belonging in the same Z500 cluster,
      %find its centroid, plot at those centroids (centroids within Z500 cluster)

%orientation  ==0 :   landscape
%             ==1,2,3,... : default (portrait)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%close all
%Coast = load('coast.mat');

figure('Name',['(EOF) Z500 Cluster Representatives',...
    'at the same date for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate Z500 eof anomalies map
    Z500_anomalies_eof=zeros(length(latitude),length(longitude));
```

```
    for j=1:eof_modes
        Z500_anomalies_eof=eof_maps_Z500(:,:,j)*pc_Z500(j,i)+Z500_anomalies_eof;
    end

    %full Z500 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    Z500_EOF=Z500_anomalies_eof+Z500_years_monthlymean(:,:,m);

    PLOT_Z500(longitude,latitude,Z500_EOF)

    %plot title %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %DateString = datestr(t_monthly(i),'dd-mmm-yyyy HH:MM:SS');
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['(EOF) Z500 Cluster #',num2str(ci(i)),',  ', DateString,', ','Population:
',num2str(population(ci(i)))],'FontSize',8);
    %xlabel('longitude (degrees North)'), ylabel('latitude (degrees East)')

    count=count+1;
end
f1=gcf;

figure('Name',['(EOF) PT2PVU at the same date as Z500 Cluster Representatives for the
month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate PT2PVU anomalies map
    PT2PVU_anomalies_eof=zeros(length(latitude),length(longitude));
    for j=1:eof_modes
        PT2PVU_anomalies_eof=eof_maps_PT2PVU(:,:,j)*pc_PT2PVU(j,i)+PT2PVU_anomalies_eof;
    end
    %full PT2PVU pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    PT2PVU_EOF=PT2PVU_anomalies_eof+PT2PVU_years_monthlymean(:,:,m);

    PLOT_PT2PVU(longitude,latitude,PT2PVU_EOF,5)

    DateString = datestr(t2(i),'mmm-yyyy');
    title(['(EOF) 2PVU PT','  ','Cluster #',num2str(ci(i)),', ', DateString],'FontSize',8);

    count=count+1;
end
f2=gcf;

figure('Name',['(EOF) W300 at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate 850hPa wind anomalies map
    map_U300=zeros(length(latitude),length(longitude)); %anomalies associated with modes
    map_V300=zeros(length(latitude),length(longitude));
    map_W300=zeros(length(latitude),length(longitude));
    for j=1:eof_modes
        map_U300=eof_maps_U300(:,:,j)*pc_U300(j,i)+map_U300;
        map_V300=eof_maps_V300(:,:,j)*pc_V300(j,i)+map_V300;
        map_W300=eof_maps_W300(:,:,j)*pc_W300(j,i)+map_W300;
    end
    %full W300 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (ex January=1)
    U300_EOF=map_U300+U300_years_monthlymean(:,:,m);
    V300_EOF=map_V300+V300_years_monthlymean(:,:,m);
    W300_EOF=map_W300+W300_years_monthlymean(:,:,m);

    PLOT_W300(longitude,latitude,U300,V300,U300_EOF,V300_EOF,W300_EOF,1)
```

```
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['(EOF) 300hPa Winds','  ','Cluster #',num2str(ci(i)),', ',
DateString],'FontSize',8);
    hold off
    count=count+1;
end
f3=gcf;

figure('Name',['(EOF) T850 at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate T850 anomalies map
    T850_anomalies_eof=zeros(length(latitude),length(longitude));
    for j=1:eof_modes
        T850_anomalies_eof=eof_maps_T850(:,:,j)*pc_T850(j,i)+T850_anomalies_eof;
    end
    %full T850 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    T850_EOF=T850_anomalies_eof+T850_years_monthlymean(:,:,m);

    PLOT_T850(longitude,latitude,T850_EOF)

    %plot title %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['(EOF) 850hPa Temp Cluster #',num2str(ci(i)),', ', DateString],'FontSize',8);

    count=count+1;
end
f4=gcf;

figure('Name',['(EOF) PV850 at the same date as Z500 Cluster Representatives for the
month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate PV850 anomalies map
    PV850_anomalies_eof=zeros(length(latitude),length(longitude));
    for j=1:eof_modes
        PV850_anomalies_eof=eof_maps_PV850(:,:,j)*pc_PV850(j,i)+PV850_anomalies_eof;
    end
    %full PV850 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    PV850_EOF=PV850_anomalies_eof+PV850_years_monthlymean(:,:,m);

    PLOT_PV850(longitude,latitude,PV850_EOF)

    DateString = datestr(t2(i),'mmm-yyyy');
    title(['(EOF) 850hPa PV (PVU)','  ','Cluster #',num2str(ci(i)),', ',
DateString],'FontSize',8);

    count=count+1;
end
f5=gcf;

figure('Name',['(EOF) W850 at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end
```

```
    %calculate 300hPa wind anomalies map
    U850_anomalies_eof=zeros(length(latitude),length(longitude));
    V850_anomalies_eof=zeros(length(latitude),length(longitude));
    W850_anomalies_eof=zeros(length(latitude),length(longitude));
    for j=1:eof_modes
        U850_anomalies_eof=eof_maps_U850(:,:,j)*pc_U850(j,i)+U850_anomalies_eof;
        V850_anomalies_eof=eof_maps_V850(:,:,j)*pc_V850(j,i)+V850_anomalies_eof;
        W850_anomalies_eof=eof_maps_W850(:,:,j)*pc_W850(j,i)+W850_anomalies_eof;
    end
    %full W850 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    U850_EOF=U850_anomalies_eof+U850_years_monthlymean(:,:,m);
    V850_EOF=V850_anomalies_eof+V850_years_monthlymean(:,:,m);
    W850_EOF=W850_anomalies_eof+W850_years_monthlymean(:,:,m);
    %OR
    %W850_EOF=sqrt(U850_EOF.^2+V850_EOF.^2); %magnitude of 850hPa wind

    PLOT_W850(longitude,latitude,U850,V850,U850_EOF,V850_EOF,W850_EOF,1)

    DateString = datestr(t2(i),'mmm-yyyy');
    title(['(EOF) 850hPa Winds','  ','Cluster #',num2str(ci(i)),', ',
DateString],'FontSize',8);

    count=count+1;
end
f6=gcf;

figure('Name',['(EOF) MSLP at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate MSLP anomalies map
    MSLP_anomalies_eof=zeros(length(latitude),length(longitude));
    for j=1:eof_modes
        MSLP_anomalies_eof=eof_maps_MSLP(:,:,j)*pc_MSLP(j,i)+MSLP_anomalies_eof;
    end

    %full MSLP pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    MSLP_EOF=MSLP_anomalies_eof+MSLP_years_monthlymean(:,:,m);

    PLOT_MSLP(longitude,latitude,MSLP_EOF,5)

    %plot title %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['(EOF) MSLP Cluster #',num2str(ci(i)),', ', DateString],'FontSize',8);

    count=count+1;
end
f7=gcf;

figure('Name',['(EOF) PV315 at the same date as Z500 Cluster Representatives for the
month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate PV315 anomalies map
    PV315_anomalies_eof=zeros(length(latitude),length(longitude));
    for j=1:eof_modes
        PV315_anomalies_eof=eof_maps_PV315(:,:,j)*pc_PV315(j,i)+PV315_anomalies_eof;
    end
    %full PV315 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    PV315_EOF=PV315_anomalies_eof+PV315_years_monthlymean(:,:,m);
```

```
    PLOT_PV315(longitude,latitude,PV315_EOF,5)

    DateString = datestr(t2(i),'mmm-yyyy');
    title(['(EOF) 315K PV (PVU)','  ','Cluster #',num2str(ci(i)),', ',
DateString],'FontSize',8);

    count=count+1;
end
f8=gcf;

figure('Name',['(EOF) SSHF at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate SSHF anomalies map
    SSHF_anomalies_eof=zeros(length(latitude),length(longitude));
    for j=1:eof_modes
        SSHF_anomalies_eof=eof_maps_SSHF(:,:,j)*pc_SSHF(j,i)+SSHF_anomalies_eof;
    end
    %full SSHF pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    SSHF_EOF=SSHF_anomalies_eof+SSHF_years_monthlymean(:,:,m);

    PLOT_SSHF(longitude, latitude, SSHF_EOF,2)

    %Title
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['(EOF) SSHF','  ','Cluster #',num2str(ci(i)),', ', DateString],'FontSize',8);

    count=count+1;
end
f9=gcf;

figure('Name',['(EOF) SLHF at the same date as Z500 Cluster Representatives for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = crti
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate SLHF eof anomalies map
    SLHF_anomalies_eof=zeros(length(latitude),length(longitude));
    for j=1:eof_modes
        SLHF_anomalies_eof=eof_maps_SLHF(:,:,j)*pc_T850(j,i)+SLHF_anomalies_eof;
    end
    %full SLHF pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (ex January=1)
    SLHF_EOF=SLHF_anomalies_eof+SLHF_years_monthlymean(:,:,m);

    PLOT_SLHF(longitude, latitude,SLHF_EOF,2)

%%% Title %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    DateString = datestr(t2(i),'mmm-yyyy');
    title(['(EOF) SLHF','  ','Cluster #',num2str(ci(i)),', ', DateString],'FontSize',8);
    %xlabel('longitude (degrees North)'), ylabel('latitude (degrees East)')
    hold off
    count=count+1;
end
f10=gcf;

%%% Save every figure as jpg picture %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if orientation==0

    set(f1, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]); %1/3 landscape screen
    set(f2, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]); %2/3 landscape screen
    set(f3, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]); %3/3 landscape screen
```

```
    set(f4, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]);
    set(f5, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]);
    set(f6, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]);

    set(f7, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]);
    set(f8, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]);
    set(f9, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]);

    set(f10, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]); %3/3 landscape screen

else
    set(f1, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f2, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f3, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f4, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f5, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f6, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f7, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f8, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f9, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f10, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %1/3 portrait screen
end

%SAVE FIGURES TO FILE
%
savename_string1 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) closest to Z500
centroid_Z500','.jpg'];
savename_string2 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) closest to Z500
centroid__PT2PVU','.jpg'];
savename_string3 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) closest to Z500
centroid__W300','.jpg'];
savename_string4 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) closest to Z500
centroid__T850','.jpg'];
savename_string5 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) closest to Z500
centroid__PV850','.jpg'];
savename_string6 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) closest to Z500
centroid__W850','.jpg'];
savename_string7 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) closest to Z500
centroid__MSLP','.jpg'];
savename_string8 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) closest to Z500
centroid__PV315','.jpg'];
savename_string9 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) closest to Z500
centroid__SSHF','.jpg'];
savename_string10 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) closest to Z500
centroid__SLHF','.jpg'];
print(f1,savename_string1, '-djpeg')
print(f2,savename_string2, '-djpeg')
print(f3,savename_string3, '-djpeg')
print(f4,savename_string4, '-djpeg')
print(f5,savename_string5, '-djpeg')
print(f6,savename_string6, '-djpeg')
print(f7,savename_string7, '-djpeg')
print(f8,savename_string8, '-djpeg')
print(f9,savename_string9, '-djpeg')
print(f10,savename_string10, '-djpeg')
%
```

The "cluster centroids of Z500"

The script "PLOT_CRM3.m" calculates and plots the cluster representatives of this case:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% K-means monthly Clustering of Z500 EOF data & of Z500 Full Patterns
CRM=3;  %cluster representatives mode:
      %1. plot the full fields closest to centroids
      %2. plot the full fields (constructing them from the EOFs and the...
                          %seasonal mean) closest to centroids (smoother)
      %3. plot the full parameter fields at Z500 cluster centroids
      %4. find the parameter PCs belonging in the same Z500 cluster,
      %find its centroid, plot at those centroids (centroids within Z500 cluster)
```

```
%orientation  ==0 :   landscape
%             ==1,2,3,... : default (portrait)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%close all
%Coast = load('coast.mat');

figure('Name',['(EOF) Z500 Centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
%Coast = load('coast.mat');
count=1;
for i=1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate Z500 eof anomalies
    Z500_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

Z500_anomalies_eof_centroid=eof_maps_Z500_norm(:,:,j)*C(i,j)+Z500_anomalies_eof_centroid;
    end

    %calculate the full Z500 patterns cluster centroids
    Z500_centroid=Z500_anomalies_eof_centroid+Z500_years_monthlymean(:,:,m);

    PLOT_Z500(longitude,latitude,Z500_centroid)

    %plot title
    title(['Z500 Cluster #',num2str(i),' ','Centroid',' ',datestr(datetime(1,m,1),'mmm'),',
','Population: ',num2str(population(i))],'FontSize',8);
    %xlabel('longitude (degrees North)'), ylabel('latitude (degrees East)')

    count=count+1; %update counter

end
f1=gcf;

figure('Name',['(EOF) PT2PVU at Z500 cluster centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PT2PVU_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

PT2PVU_anomalies_eof_centroid=eof_maps_PT2PVU_norm(:,:,j)*C(i,j)+PT2PVU_anomalies_eof_centroid
;
    end
    %full PT2PVU pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    PT2PVU_centroid=PT2PVU_anomalies_eof_centroid+PT2PVU_years_monthlymean(:,:,m);

    PLOT_PT2PVU(longitude,latitude,PT2PVU_centroid,5)

    %plot title
    title(['2PVU PT at Z500 Cluster #',num2str(i),' ','Centroid','
',datestr(datetime(1,m,1),'mmm')]);

    count=count+1;
end
f2=gcf;

figure('Name',['W300 at Z500 cluster centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
```

```
        subplot(k,1,count);
    end

    %calculate 300hPa anomalies components at Z500 cluster centroids
    U300_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    V300_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    W300_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

U300_anomalies_eof_centroid=eof_maps_U300_norm(:,:,j)*C(i,j)+U300_anomalies_eof_centroid;

V300_anomalies_eof_centroid=eof_maps_V300_norm(:,:,j)*C(i,j)+V300_anomalies_eof_centroid;

W300_anomalies_eof_centroid=eof_maps_W300_norm(:,:,j)*C(i,j)+W300_anomalies_eof_centroid;
    end

    %full W300 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (ex January=1)
    U300_centroid=U300_anomalies_eof_centroid+U300_years_monthlymean(:,:,m);
    V300_centroid=V300_anomalies_eof_centroid+V300_years_monthlymean(:,:,m);
    W300_centroid=W300_anomalies_eof_centroid+W300_years_monthlymean(:,:,m);

    PLOT_W300(longitude,latitude,U300,V300,U300_centroid,V300_centroid,W300_centroid,1)

    %plot title
    title(['300hPa Winds at Z500 Cluster #',num2str(i),' ','Centroid','
',datestr(datetime(1,m,1),'mmm')]);

    count=count+1;
end
f3=gcf;

figure('Name',['(EOF) T850 at Z500 cluster centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    T850_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

T850_anomalies_eof_centroid=eof_maps_T850_norm(:,:,j)*C(i,j)+T850_anomalies_eof_centroid;
    end
    %full T850 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    T850_centroid=T850_anomalies_eof_centroid+T850_years_monthlymean(:,:,m);

    PLOT_T850(longitude,latitude,T850_centroid)

    %title
    title(['T850 at Z500 Cluster #',num2str(i),' ','Centroid','
',datestr(datetime(1,m,1),'mmm')]);

    count=count+1;
end
f4=gcf;

figure('Name',['(EOF) PV850 at Z500 cluster Centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PV850_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

PV850_anomalies_eof_centroid=eof_maps_PV850_norm(:,:,j)*C(i,j)+PV850_anomalies_eof_centroid;
    end
    %full PV850 pattern may be constructed by the anomalous field constructed
```

```
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    PV850_centroid=PV850_anomalies_eof_centroid+PV850_years_monthlymean(:,:,m);

    PLOT_PV850(longitude,latitude,PV850_centroid)

    title(['850hPa PV at Z500 Cluster #',num2str(i),' ','Centroid','
',datestr(datetime(1,m,1),'mmm')]);

    count=count+1;
end
f5=gcf;

figure('Name',['(EOF) W850 at Z500 cluster Centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    U850_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    V850_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    W850_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

U850_anomalies_eof_centroid=eof_maps_U850_norm(:,:,j)*C(i,j)+U850_anomalies_eof_centroid;

V850_anomalies_eof_centroid=eof_maps_V850_norm(:,:,j)*C(i,j)+V850_anomalies_eof_centroid;

W850_anomalies_eof_centroid=eof_maps_W850_norm(:,:,j)*C(i,j)+W850_anomalies_eof_centroid;
    end
    %full W850 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    U850_centroid=U850_anomalies_eof_centroid+U850_years_monthlymean(:,:,m);
    V850_centroid=V850_anomalies_eof_centroid+V850_years_monthlymean(:,:,m);
    W850_centroid=W850_anomalies_eof_centroid+W850_years_monthlymean(:,:,m);
    %OR
    %W850_centroid=sqrt(U850_centroid.^2+V850_centroid.^2); %magnitude of 850hPa wind

    PLOT_W850(longitude,latitude,U850,V850,U850_centroid,V850_centroid,W850_centroid,1)

    title(['850hPa Winds at Z500 Cluster #',num2str(i),' ','Centroid','
',datestr(datetime(1,m,1),'mmm')]);

    count=count+1;
end
f6=gcf;

figure('Name',['(EOF) MSLP at Z500 cluster Centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    MSLP_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

MSLP_anomalies_eof_centroid=eof_maps_MSLP_norm(:,:,j)*C(i,j)+MSLP_anomalies_eof_centroid;
    end

    %full MSLP pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    MSLP_centroid=MSLP_anomalies_eof_centroid+MSLP_years_monthlymean(:,:,m);

    PLOT_MSLP(longitude,latitude,MSLP_centroid,5)

    %plot title
    title(['MSLP at Z500 Cluster #',num2str(i),' ','Centroid','
',datestr(datetime(1,m,1),'mmm')]);

    count=count+1;
```

```
end
f7=gcf;

figure('Name',['(EOF) PV315 at Z500 cluster Centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    PV315_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

PV315_anomalies_eof_centroid=eof_maps_PV315_norm(:,:,j)*C(i,j)+PV315_anomalies_eof_centroid;
    end
    %full PV315 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    PV315_centroid=PV315_anomalies_eof_centroid+PV315_years_monthlymean(:,:,m);

    PLOT_PV315(longitude,latitude,PV315_centroid,5)

    title(['315K PV at Z500 Cluster #',num2str(i),' ','Centroid','
',datestr(datetime(1,m,1),'mmm')]);

    count=count+1;
end
f8=gcf;

figure('Name',['(EOF) SSHF at Z500 cluster Centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    SSHF_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

SSHF_anomalies_eof_centroid=eof_maps_SSHF_norm(:,:,j)*C(i,j)+SSHF_anomalies_eof_centroid;
    end
    %full SSHF pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    SSHF_centroid=SSHF_anomalies_eof_centroid+SSHF_years_monthlymean(:,:,m);

    PLOT_SSHF(longitude, latitude, SSHF_centroid,2)

    %Title
    title(['SSHF at Z500 Cluster #',num2str(i),' ','Centroid','
',datestr(datetime(1,m,1),'mmm')]);

    count=count+1;
end
f9=gcf;

figure('Name',['(EOF) SLHF at Z500 cluster Centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %calculate SLHF eof anomalies
    SLHF_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

SLHF_anomalies_eof_centroid=eof_maps_SLHF_norm(:,:,j)*C(i,j)+SLHF_anomalies_eof_centroid;
    end
```

```
    %calculate SLHF patterns cluster of Z500 centroids
    SLHF_centroid=SLHF_anomalies_eof_centroid+SLHF_years_monthlymean(:,:,m);

    PLOT_SLHF(longitude, latitude,SLHF_centroid,2)

%%% Title
    title(['SLHF at Z500 Cluster #',num2str(i),' ','Centroid','
',datestr(datetime(1,m,1),'mmm')]);

    count=count+1;
end
f10=gcf;

%%% Save every figure as jpg picture %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if orientation==0

    set(f1, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]); %1/3 landscape screen
    set(f2, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]); %2/3 landscape screen
    set(f3, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]); %3/3 landscape screen

    set(f4, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]);
    set(f5, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]);
    set(f6, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]);

    set(f7, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]);
    set(f8, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]);
    set(f9, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]);

    set(f10, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]); %3/3 landscape screen

else
    set(f1, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f2, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f3, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f4, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f5, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f6, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f7, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f8, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f9, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f10, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %1/3 portrait screen
end

%SAVE FIGURES TO FILE
%
savename_string1 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) Z500 centroid','.jpg'];
savename_string2 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)PT2PVU at Z500
centroid','.jpg'];
savename_string3 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)W300 at Z500 centroid','.jpg'];
savename_string4 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)T850 at Z500 centroid','.jpg'];
savename_string5 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)PV850 at Z500
centroid','.jpg'];
savename_string6 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)W850 at Z500 centroid','.jpg'];
savename_string7 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)MSLP at Z500 centroid','.jpg'];
savename_string8 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)PV315 at Z500
centroid','.jpg'];
savename_string9 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)SSHF at Z500 centroid','.jpg'];
savename_string10 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)SLHF at Z500
centroid','.jpg'];
print(f1,savename_string1, '-djpeg')
print(f2,savename_string2, '-djpeg')
print(f3,savename_string3, '-djpeg')
print(f4,savename_string4, '-djpeg')
print(f5,savename_string5, '-djpeg')
print(f6,savename_string6, '-djpeg')
print(f7,savename_string7, '-djpeg')
print(f8,savename_string8, '-djpeg')
print(f9,savename_string9, '-djpeg')
print(f10,savename_string10, '-djpeg')
%
```

The "cluster centroids of each parameter"

The script "PLOT_CRM4.m" calculates and plots the cluster representatives of this case:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% K-means monthly Clustering of Z500 EOF data & of Z500 Full Patterns
CRM=4;  %cluster representatives mode:
      %1. plot the full fields closest to centroids
      %2. plot the full fields (constructing them from the EOFs and the...
                              %seasonal mean) closest to centroids (smoother)
      %3. plot the full parameter fields at Z500 cluster centroids
      %4. find the parameter PCs belonging in the same Z500 cluster,
      %find its centroid, plot at those centroids (centroids within Z500 cluster)

%orientation  ==0 :   landscape
%             ==1,2,3,... : default (portrait)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

C_PT2PVU=zeros(k,eof_modes); %PT2PVU PCs centroids belonging in the same Z500 cluster
C_U300=zeros(k,eof_modes); %U300 PCs centroids belonging to the same Z500 cluster
C_V300=zeros(k,eof_modes); %V300 PCs centroids belonging to the same Z500 cluster
C_W300=zeros(k,eof_modes); %W300 PCs centroids belonging to the same Z500 cluster
C_T850=zeros(k,eof_modes); %T850 PCs centroids belonging in the same Z500 cluster
C_PV850=zeros(k,eof_modes); %PV850 PCs centroids belonging in the same Z500 cluster
C_U850=zeros(k,eof_modes); %U850 PCs centroids belonging in the same Z500 cluster
C_V850=zeros(k,eof_modes); %V850 PCs centroids belonging in the same Z500 cluster
C_W850=zeros(k,eof_modes); %W850 PCs centroids belonging in the same Z500 cluster
C_MSLP=zeros(k,eof_modes); %MSLP PCs centroids belonging in the same Z500 cluster
C_PV315=zeros(k,eof_modes); %PV315 PCs centroids belonging in the same Z500 cluster
C_SSHF=zeros(k,eof_modes); %SSHF PCs centroids belonging to the same Z500 cluster
C_SLHF=zeros(k,eof_modes); %SLHF PCs centroids belonging to the same Z500 cluster

%close all
%Coast = load('coast.mat');

figure('Name',['(EOF) Z500 Centroids for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    %plot Z500 isolines centroids %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    Z500_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

Z500_anomalies_eof_centroid=eof_maps_Z500_norm(:,:,j)*C(i,j)+Z500_anomalies_eof_centroid;
    end

    %calculate the full Z500 patterns cluster centroids
    Z500_centroid=Z500_anomalies_eof_centroid+Z500_years_monthlymean(:,:,m);

    PLOT_Z500(longitude,latitude,Z500_centroid)

    %plot title
    title(['Z500 Cluster #',num2str(i),' ','Centroid',', ',datestr(datetime(1,m,1),'mmm'),',
','Population: ',num2str(population(i))],'FontSize',8);
    %xlabel('longitude (degrees North)'), ylabel('latitude (degrees East)')

    count=count+1; %update counter

end
f1=gcf;

figure('Name',['(EOF) PT2PVU centroids within the same Z500 cluster for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end
```

```
    A=find(idx==i);
    B=index(A);     %find time steps belonging to cluster 'i'

    %PCs centroids belonging in the same Z500 cluster
    [idx_PT2PVU,K] = kmeans(pc_PT2PVU_norm(1:eof_modes,B)',1);
    C_PT2PVU(i,:)=K;

    PT2PVU_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

PT2PVU_anomalies_eof_centroid=eof_maps_PT2PVU_norm(:,:,j)*C_PT2PVU(i,j)+PT2PVU_anomalies_eof_c
entroid;
    end
    %full PT2PVU pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    PT2PVU_centroid=PT2PVU_anomalies_eof_centroid+PT2PVU_years_monthlymean(:,:,m);

    PLOT_PT2PVU(longitude,latitude,PT2PVU_centroid,5)

    title(['2PVU PT Centroid of Z500 Cluster #',num2str(i),',
',datestr(datetime(1,m,1),'mmm')],'FontSize',8);

    count=count+1;
end
f2=gcf;

figure('Name',['(EOF) W300 centroids within the same Z500 cluster for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    A=find(idx==i);
    B=index(A);     %find time steps belonging to cluster 'i'

    %calculate 300hPa anomalies components (u,v,w) EOF MAPS at their
    %PCs centroids belonging to the same Z500 cluster %%%%%%%%%%%%%
    [idx_U300,K] = kmeans(pc_U300_norm(1:eof_modes,B)',1);
    C_U300(i,:)=K;
    [idx_V300,K] = kmeans(pc_V300_norm(1:eof_modes,B)',1);
    C_V300(i,:)=K;
    [idx_W300,K] = kmeans(pc_W300_norm(1:eof_modes,B)',1);
    C_W300(i,:)=K;

    %calculate 300hPa anomalies components at Z500 cluster centroids
    U300_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    V300_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    W300_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

U300_anomalies_eof_centroid=eof_maps_U300_norm(:,:,j)*C_U300(i,j)+U300_anomalies_eof_centroid;

V300_anomalies_eof_centroid=eof_maps_V300_norm(:,:,j)*C_V300(i,j)+V300_anomalies_eof_centroid;

W300_anomalies_eof_centroid=eof_maps_W300_norm(:,:,j)*C_W300(i,j)+W300_anomalies_eof_centroid;
    end

    %full W300 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (ex January=1)
    U300_centroid=U300_anomalies_eof_centroid+U300_years_monthlymean(:,:,m);
    V300_centroid=V300_anomalies_eof_centroid+V300_years_monthlymean(:,:,m);
    W300_centroid=W300_anomalies_eof_centroid+W300_years_monthlymean(:,:,m);

    PLOT_W300(longitude,latitude,U300,V300,U300_centroid,V300_centroid,W300_centroid,1)

    title(['300hPa Winds Centroid of Z500 Cluster #',num2str(i),',
',datestr(datetime(1,m,1),'mmm')],'FontSize',8);
    hold off
    count=count+1;
end
f3=gcf;
```

```
figure('Name',['(EOF) T850 centroids within the same Z500 cluster for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    A=find(idx==i);
    B=index(A);     %find time steps belonging to cluster 'i'

    %PCs centroids belonging in the same Z500 cluster
    [idx_T850,K] = kmeans(pc_T850_norm(1:eof_modes,B)',1);
    C_T850(i,:)=K;

    T850_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

T850_anomalies_eof_centroid=eof_maps_T850_norm(:,:,j)*C_T850(i,j)+T850_anomalies_eof_centroid;
    end
    %full T850 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    T850_centroid=T850_anomalies_eof_centroid+T850_years_monthlymean(:,:,m);

    PLOT_T850(longitude,latitude,T850_centroid)

    %title
    title(['T850 Centroid of Z500 Cluster #',num2str(i),',
',datestr(datetime(1,m,1),'mmm')],'FontSize',8);

    count=count+1;
end
f4=gcf;

figure('Name',['(EOF) PV850 centroids within the same Z500 cluster for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    A=find(idx==i);
    B=index(A);     %find time steps belonging to cluster 'i'

    %PCs centroids belonging in the same Z500 cluster
    [idx_PV850,K] = kmeans(pc_PV850_norm(1:eof_modes,B)',1);
    C_PV850(i,:)=K;

    PV850_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

PV850_anomalies_eof_centroid=eof_maps_PV850_norm(:,:,j)*C_PV850(i,j)+PV850_anomalies_eof_centr
oid;
    end
    %full PV850 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    PV850_centroid=PV850_anomalies_eof_centroid+PV850_years_monthlymean(:,:,m);

    PLOT_PV850(longitude,latitude,PV850_centroid)

    title(['850hPa PV Centroid of Z500 Cluster #',num2str(i),',
',datestr(datetime(1,m,1),'mmm')],'FontSize',8);

    count=count+1;
end
f5=gcf;

figure('Name',['(EOF) W850 centroids within the same Z500 cluster for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
```

```
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    A=find(idx==i);
    B=index(A);     %find time steps belonging to cluster 'i'

    %PCs centroids belonging in the same Z500 cluster
    [idx_U850,K] = kmeans(pc_U850_norm(1:eof_modes,B)',1);
    C_U850(i,:)=K;
    [idx_V850,K] = kmeans(pc_V850_norm(1:eof_modes,B)',1);
    C_V850(i,:)=K;
    [idx_W850,K] = kmeans(pc_W850_norm(1:eof_modes,B)',1);
    C_W850(i,:)=K;

    U850_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    V850_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    W850_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

U850_anomalies_eof_centroid=eof_maps_U850_norm(:,:,j)*C_U850(i,j)+U850_anomalies_eof_centroid;

V850_anomalies_eof_centroid=eof_maps_V850_norm(:,:,j)*C_V850(i,j)+V850_anomalies_eof_centroid;

W850_anomalies_eof_centroid=eof_maps_W850_norm(:,:,j)*C_W850(i,j)+W850_anomalies_eof_centroid;
    end

    %full W850 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    U850_centroid=U850_anomalies_eof_centroid+U850_years_monthlymean(:,:,m);
    V850_centroid=V850_anomalies_eof_centroid+V850_years_monthlymean(:,:,m);
    W850_centroid=W850_anomalies_eof_centroid+W850_years_monthlymean(:,:,m);
    %OR
    %W850_centroid=sqrt(U850_centroid.^2+V850_centroid.^2); %magnitude of 850hPa wind

    PLOT_W850(longitude,latitude,U850,V850,U850_centroid,V850_centroid,W850_centroid,1)

    title(['850hPa Winds Centroid of Z500 Cluster #',num2str(i),',
',datestr(datetime(1,m,1),'mmm')],'FontSize',8);

    count=count+1;
end
f6=gcf;

figure('Name',['(EOF) MSLP centroids within the same Z500 cluster for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    A=find(idx==i);
    B=index(A);     %find time steps belonging to cluster 'i'

    %PCs centroids belonging in the same Z500 cluster
    [idx_MSLP,K] = kmeans(pc_MSLP_norm(1:eof_modes,B)',1);
    C_MSLP(i,:)=K;

    MSLP_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

MSLP_anomalies_eof_centroid=eof_maps_MSLP_norm(:,:,j)*C_MSLP(i,j)+MSLP_anomalies_eof_centroid;
    end

    %full MSLP pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    MSLP_centroid=MSLP_anomalies_eof_centroid+MSLP_years_monthlymean(:,:,m);

    PLOT_MSLP(longitude,latitude,MSLP_centroid,5)

    %plot title
    title(['MSLP Centroid of Z500 Cluster #',num2str(i),',
',datestr(datetime(1,m,1),'mmm')],'FontSize',8);
```

```
    count=count+1;
end
f7=gcf;

figure('Name',['(EOF) PV315 centroids within the same Z500 cluster for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    A=find(idx==i);
    B=index(A);     %find time steps belonging to cluster 'i'

    %PCs centroids belonging in the same Z500 cluster
    [idx_PV315,K] = kmeans(pc_PV315_norm(1:eof_modes,B)',1);
    C_PV315(i,:)=K;

    PV315_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

PV315_anomalies_eof_centroid=eof_maps_PV315_norm(:,:,j)*C_PV315(i,j)+PV315_anomalies_eof_centr
oid;
    end
    %full PV315 pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    PV315_centroid=PV315_anomalies_eof_centroid+PV315_years_monthlymean(:,:,m);

    PLOT_PV315(longitude,latitude,PV315_centroid,5)

    title(['315K PV (PVU) Centroid of Z500 Cluster #',num2str(i),',
',datestr(datetime(1,m,1),'mmm')],'FontSize',8);

    count=count+1;
end
f8=gcf;

figure('Name',['(EOF) SSHF centroids within the same Z500 cluster for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i = 1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    A=find(idx==i);
    B=index(A);     %find time steps belonging to cluster 'i'

    %PCs centroids belonging in the same Z500 cluster
    [idx_SSHF,K] = kmeans(pc_SSHF_norm(1:eof_modes,B)',1);
    C_SSHF(i,:)=K;

    SSHF_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

SSHF_anomalies_eof_centroid=eof_maps_SSHF_norm(:,:,j)*C_SSHF(i,j)+SSHF_anomalies_eof_centroid;
    end
    %full SSHF pattern may be constructed by the anomalous field constructed
    %by the EOFs "plus" the "seasonal mean of that month" (January=1)
    SSHF_centroid=SSHF_anomalies_eof_centroid+SSHF_years_monthlymean(:,:,m);

    PLOT_SSHF(longitude, latitude, SSHF_centroid,2)

    %Title
    title(['SSHF Centroid of Z500 Cluster #',num2str(i),',
',datestr(datetime(1,m,1),'mmm')],'FontSize',8);

    count=count+1;
end
f9=gcf;
```

```
figure('Name',['(EOF) SLHF centroids within the same Z500 cluster for the month',...
    '  ', datestr(datetime(1,m,1),'mmmm')],'NumberTitle','off');
count=1;
for i=1:k
    if orientation==0 % landscape
        subplot(1,k,count);
    else % portrait / default
        subplot(k,1,count);
    end

    A=find(idx==i);
    B=index(A);     %find time steps belonging to cluster 'i'

    %PCs centroids belonging in the same Z500 cluster
    [idx_SLHF,K] = kmeans(pc_SLHF_norm(1:eof_modes,B)',1);
    C_SLHF(i,:)=K;

    %plot SLHF pcolor at Z500 centroids
    SLHF_anomalies_eof_centroid=zeros(length(latitude),length(longitude));
    for j=1:eof_modes

SLHF_anomalies_eof_centroid=eof_maps_SLHF_norm(:,:,j)*C_SLHF(i,j)+SLHF_anomalies_eof_centroid;
    end


    %calculate SLHF patterns cluster of Z500 centroids
    SLHF_centroid=SLHF_anomalies_eof_centroid+SLHF_years_monthlymean(:,:,m);

    PLOT_SLHF(longitude, latitude,SLHF_centroid,2)

    %Title
    title(['SLHF Centroid of Z500 Cluster #',num2str(i),',
',datestr(datetime(1,m,1),'mmm')],'FontSize',8);

    count=count+1;
end
f10=gcf;

%%% Save every figure as jpg picture %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if orientation==0

    set(f1, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]); %1/3 landscape screen
    set(f2, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]); %2/3 landscape screen
    set(f3, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]); %3/3 landscape screen

    set(f4, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]);
    set(f5, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]);
    set(f6, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]);

    set(f7, 'Units', 'Normalized', 'OuterPosition', [0 2/3 1 1/3]);
    set(f8, 'Units', 'Normalized', 'OuterPosition', [0 1/3 1 1/3]);
    set(f9, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]);

    set(f10, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1/3]); %3/3 landscape screen

else
    set(f1, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f2, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f3, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f4, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f5, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f6, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f7, 'Units', 'Normalized', 'OuterPosition', [0 0 .33 1]); %1/3 portrait screen
    set(f8, 'Units', 'Normalized', 'OuterPosition', [.33 0 .33 1]); %2/3 portrait screen
    set(f9, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %3/3 portrait screen

    set(f10, 'Units', 'Normalized', 'OuterPosition', [.66 0 .33 1]); %1/3 portrait screen
end

%SAVE FIGURES TO FILE
%
savename_string1 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF) Z500 centroid','.jpg'];
savename_string2 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)PT2PVU centroid','.jpg'];
savename_string3 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)W300 centroid','.jpg'];
```

```
savename_string4 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)T850 centroid','.jpg'];
savename_string5 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)PV850 centroid','.jpg'];
savename_string6 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)W850 centroid','.jpg'];
savename_string7 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)MSLP centroid','.jpg'];
savename_string8 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)PV315 centroid','.jpg'];
savename_string9 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)SSHF centroid','.jpg'];
savename_string10 = ['C:\datestr(datetime(1,m,1),'mmmm'),'_(EOF)SLHF centroid','.jpg'];
print(f1,savename_string1, '-djpeg')
print(f2,savename_string2, '-djpeg')
print(f3,savename_string3, '-djpeg')
print(f4,savename_string4, '-djpeg')
print(f5,savename_string5, '-djpeg')
print(f6,savename_string6, '-djpeg')
print(f7,savename_string7, '-djpeg')
print(f8,savename_string8, '-djpeg')
print(f9,savename_string9, '-djpeg')
print(f10,savename_string10, '-djpeg')
%
```

Choosing the "best" cluster representatives

For EOF modes = [2 5 10 20 50] we find and plot the mean Euclidean errors (section 1.2.7):

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Clustering of the EOF data & of Full Patterns
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
k=4;                          %Number of Clusters to divide data
Replicates=3000;
CRM=2; %cluster representatives mode:
      %1. plot the full fields closest to centroids
      %2. plot the full fields (constructing them from the EOFs and the...
                              %seasonal mean) closest to centroids (smoother)
      %3. plot the full Z500 cluster centroids
ci=zeros(length(t2),1);      %Vector with cluster indices

er3_Z500=zeros(1,12);
er4_Z500=zeros(1,12);

er1_MSLP=zeros(1,12);
er2_MSLP=zeros(1,12);
er3_MSLP=zeros(1,12);
er4_MSLP=zeros(1,12);

er1_T850=zeros(1,12);
er2_T850=zeros(1,12);
er3_T850=zeros(1,12);
er4_T850=zeros(1,12);

er1_SLHF=zeros(1,12);
er2_SLHF=zeros(1,12);
er3_SLHF=zeros(1,12);
er4_SLHF=zeros(1,12);

er1_W300=zeros(1,12);
er2_W300=zeros(1,12);
er3_W300=zeros(1,12);
er4_W300=zeros(1,12);

er1_PV850=zeros(1,12);
er2_PV850=zeros(1,12);
er3_PV850=zeros(1,12);
er4_PV850=zeros(1,12);

er1_W850=zeros(1,12);
er2_W850=zeros(1,12);
er3_W850=zeros(1,12);
er4_W850=zeros(1,12);

er1_SSHF=zeros(1,12);
er2_SSHF=zeros(1,12);
er3_SSHF=zeros(1,12);
er4_SSHF=zeros(1,12);

er1_PV315=zeros(1,12);
```

```
er2_PV315=zeros(1,12);
er3_PV315=zeros(1,12);
er4_PV315=zeros(1,12);

er1_PT2PVU=zeros(1,12);
er2_PT2PVU=zeros(1,12);
er3_PT2PVU=zeros(1,12);
er4_PT2PVU=zeros(1,12);

count=11;
for eof_modes=[2 5 10 20 50]

% loop for each unique month %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for m=min(month2):max(month2)

    index= find(month2==m); %month position index in t2 array

%%% Z500 based Kmeans clustering %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    pc_month1_Z500=pc_Z500_norm(1:eof_modes,index)';
    [idx,C,sumd,D] = kmeans(pc_month1_Z500,k,'Replicates',Replicates);
    disp(' ');
    disp(['The Best total sum of K-mean distances = ',num2str(sum(sumd))]);

%%% Clustering Representatives (closest to Z500 cluster centroids) %%%%%%%%
    minD=min(D); %row vector containing the min of each column of D
    [row,col] = find(D==minD); % row: returns the "time steps" of the best representative
                               % col: returns the cluster it belongs in
    %
    %in case multiple distances of columns of D (same cluster) are equal
    %keep only the fist occurrence and its index
    [ucol,ui]=unique(col); % ucol: vector the unique clusters same as unique(col)
                           % ui:   vector with the unique cluster indices
    row_unique=row(ui);  % unique D "time steps"
    crti=index(row_unique);     % cluster reference time indices in t2 array
    ci(index)=idx;              % update the cluster indices (ci) vector
    datestr(t2(crti))           % to see the dates of min RMSD

%%% MSLP centroids of points belonging to same Z500 cluster %%%%%%%%%%%%%%%%
    C_MSLP=zeros(k,eof_modes); %MSLP centroids by Z500 clustering
    for i=1:k
        A=find(idx==i); %find PCs in the i-th Z500 cluster
        B=index(A);
        [idx_MSLP,K] = kmeans(pc_MSLP_norm(1:eof_modes,B)',1); %kmeans with 1 cluster(to find
centroids matrix K)
        C_MSLP(i,:)=K; % (k,eof_modes) matrix centroids of PCs
                                    %(that belong in the same Z500 cluster)
    end

%%% T850 centroids of points belonging to same Z500 cluster %%%%%%%%%%%%%%%%
    C_T850=zeros(k,eof_modes);
    for i=1:k
        A=find(idx==i);
        B=index(A);
        [idx_T850,K] = kmeans(pc_T850_norm(1:eof_modes,B)',1);
        C_T850(i,:)=K; % (k,eof_modes) matrix centroids of PCs
                                    %(that belong in the same Z500 cluster)
    end

%%% SLHF centroids of points belonging to same Z500 cluster %%%%%%%%%%%%%%%%
    C_SLHF=zeros(k,eof_modes);
    for i=1:k
        A=find(idx==i);
        B=index(A);
        [idx_SLHF,K] = kmeans(pc_SLHF_norm(1:eof_modes,B)',1);
        C_SLHF(i,:)=K; % (k,eof_modes) matrix centroids of PCs
                                    %(that belong in the same Z500 cluster)
    end

%%% W300 centroids of points belonging to same Z500 cluster %%%%%%%%%%%%%%%%
    C_W300=zeros(k,eof_modes);
    for i=1:k
        A=find(idx==i);
        B=index(A);
        [idx_W300,K] = kmeans(pc_W300_norm(1:eof_modes,B)',1);
        C_W300(i,:)=K; % (k,eof_modes) matrix centroids of PCs
                                    %(that belong in the same Z500 cluster)
    end
```

```
%%% PV850 centroids of points belonging to same Z500 cluster %%%%%%%%%%%%%%%
    C_PV850=zeros(k,eof_modes);
    for i=1:k
        A=find(idx==i);
        B=index(A);
        [idx_PV850,K] = kmeans(pc_PV850_norm(1:eof_modes,B)',1);
        C_PV850(i,:)=K; % (k,eof_modes) matrix centroids of PCs
                                    %(that belong in the same Z500 cluster)
    end

%%% W850 centroids of points belonging to same Z500 cluster %%%%%%%%%%%%%%%
    C_W850=zeros(k,eof_modes);
    for i=1:k
        A=find(idx==i);
        B=index(A);
        [idx_W850,K] = kmeans(pc_W850_norm(1:eof_modes,B)',1);
        C_W850(i,:)=K; % (k,eof_modes) matrix centroids of PCs
                                    %(that belong in the same Z500 cluster)
    end

%%% SSHF centroids of points belonging to same Z500 cluster %%%%%%%%%%%%%%%
    C_SSHF=zeros(k,eof_modes);
    for i=1:k
        A=find(idx==i);
        B=index(A);
        [idx_SSHF,K] = kmeans(pc_SSHF_norm(1:eof_modes,B)',1);
        C_SSHF(i,:)=K; % (k,eof_modes) matrix centroids of PCs
                                    %(that belong in the same Z500 cluster)
    end

%%% PV315 centroids of points belonging to same Z500 cluster %%%%%%%%%%%%%%%
    C_PV315=zeros(k,eof_modes);
    for i=1:k
        A=find(idx==i);
        B=index(A);
        [idx_PV315,K] = kmeans(pc_PV315_norm(1:eof_modes,B)',1);
        C_PV315(i,:)=K; % (k,eof_modes) matrix centroids of PCs
                                    %(that belong in the same Z500 cluster)
    end

%%% PT2PVU centroids of points belonging to same Z500 cluster %%%%%%%%%%%%%%%
    C_PT2PVU=zeros(k,eof_modes);
    for i=1:k
        A=find(idx==i);
        B=index(A);
        [idx_PT2PVU,K] = kmeans(pc_PT2PVU_norm(1:eof_modes,B)',1);
        C_PT2PVU(i,:)=K; % (k,eof_modes) matrix centroids of PCs
                                    %(that belong in the same Z500 cluster)
    end

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %FIND the combination within cluster of representative that corresponds
    %to the minimum distance from cluster centers %%%%%%%%%%%%%%%%%%%%%%%%%
    D_MSLP=zeros(length(index),k); %square L2 distances of all MSLP PCs from centroids of MSLP
PCs
    D_T850=zeros(length(index),k);
    D_SLHF=zeros(length(index),k);
    D_W300=zeros(length(index),k);

    D_PV850=zeros(length(index),k);
    D_W850=zeros(length(index),k);
    D_SSHF=zeros(length(index),k);
    D_PV315=zeros(length(index),k);
    D_PT2PVU=zeros(length(index),k);
    for i=1:k
        sq_MSLP=zeros(length(index),1); %Squared Euclidean distances between each MSLP PC -&-
centroid of MSLP PCs within i-th Z500 cluster
        sq_T850=zeros(length(index),1);
        sq_SLHF=zeros(length(index),1);
        sq_W300=zeros(length(index),1);

        sq_PV850=zeros(length(index),1);
        sq_W850=zeros(length(index),1);
        sq_SSHF=zeros(length(index),1);
        sq_PV315=zeros(length(index),1);
        sq_PT2PVU=zeros(length(index),1);
```

```
        for j=1:eof_modes
            sq_MSLP=(pc_MSLP_norm(j,index)'-C_MSLP(i,j)).^2+sq_MSLP;
            sq_T850=(pc_T850_norm(j,index)'-C_T850(i,j)).^2+sq_T850;
            sq_SLHF=(pc_SLHF_norm(j,index)'-C_SLHF(i,j)).^2+sq_SLHF;
            sq_W300=(pc_W300_norm(j,index)'-C_W300(i,j)).^2+sq_W300;

            sq_PV850=(pc_PV850_norm(j,index)'-C_PV850(i,j)).^2+sq_PV850;
            sq_W850=(pc_W850_norm(j,index)'-C_W850(i,j)).^2+sq_W850;
            sq_SSHF=(pc_SSHF_norm(j,index)'-C_SSHF(i,j)).^2+sq_SSHF;
            sq_PV315=(pc_PV315_norm(j,index)'-C_PV315(i,j)).^2+sq_PV315;
            sq_PT2PVU=(pc_PT2PVU_norm(j,index)'-C_PT2PVU(i,j)).^2+sq_PT2PVU;
        end
        D_MSLP(:,i)=(sq_MSLP);
        D_T850(:,i)=(sq_MSLP);
        D_SLHF(:,i)=(sq_MSLP);
        D_W300(:,i)=(sq_MSLP);

        D_PV850(:,i)=(sq_PV850);
        D_W850(:,i)=(sq_W850);
        D_SSHF(:,i)=(sq_SSHF);
        D_PV315(:,i)=(sq_PV315);
        D_PT2PVU(:,i)=(sq_PT2PVU);
    end
    D_total=D+D_MSLP+D_T850+D_SLHF+D_W300+D_PV850+D_W850+D_SSHF+D_PV315+D_PT2PVU; %sum of all
Squared Euclidean distances from centroids

    %select points that belong in the same cluster (in case the total best...
    %representative wants to jump to another cluster)
    row_unique1=zeros(k,1);
    for i=1:k
        A=find(idx==i);
        [M,I]=min(D_total(A,i)); %M:the min of each column of "D_total" in cluster i
                                 %I:index of the first occurrence of min
        row_unique1(i)=A(I); %pick the first occurrence in column i
    end

    crti1=index(row_unique1);     % cluster reference time indices in t2 array
    ci(crti1) %show the clusters they belong in (should be 1,2,3,4,...k)

%%% Z500 PCs/Centroids/Representatives/Best total representative..
%(2D) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    sq3=zeros(k,1); %Squared Euclidean distances between Z500 cluster centroids and Z500
representatives
    sq4=zeros(k,1); %Squared Euclidean distances between Z500 cluster centroids and Z500 total
best representative
    for j=1:eof_modes
        sq3=(C(:,j)-pc_Z500_norm(j,crti)').^2+sq3;
        sq4=(C(:,j)-pc_Z500_norm(j,crti1)').^2+sq4;
    end

    er3_Z500(m)=sum(sqrt(sq3))/k; %square mean root of sq3 of month m
    er4_Z500(m)=sum(sqrt(sq4))/k; %square mean root of sq4 of month m

%%% MSLP PCs/Centroids of PCs belonging to the same Z500 cluster/...
% Z500 centroids/PCs same as Z500 representatives/...
% PCs same as Z500 Best total representative %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    sq1=zeros(k,1); %Squared Euclidean distances between Parameter cluster centroids and Z500
cluster centroids
    sq2=zeros(k,1); %Squared Euclidean distances between Parameter cluster centroids and
Parameter PCs same as Z500 representatives
    sq3=zeros(k,1); %Squared Euclidean distances between Z500 cluster centroids and Parameter
PCs same as Z500 representatives
    sq4=zeros(k,1); %Squared Euclidean distances between Parameter cluster centroids and
Parameter PCs same as Z500 total best representative
    for j=1:eof_modes
        sq1=(C_MSLP(:,j)-C(:,j)).^2+sq1;
        sq2=(C_MSLP(:,j)-pc_MSLP_norm(j,crti)').^2+sq2;
        sq3=(C(:,j)-pc_MSLP_norm(j,crti)').^2+sq3;
        sq4=(C_MSLP(:,j)-pc_MSLP_norm(j,crti1)').^2+sq4;
    end

    er1_MSLP(m)=sum(sqrt(sq1))/k; %square mean root of sq1 of month m
    er2_MSLP(m)=sum(sqrt(sq2))/k; %square mean root of sq2 of month m
    er3_MSLP(m)=sum(sqrt(sq3))/k; %square mean root of sq3 of month m
    er4_MSLP(m)=sum(sqrt(sq4))/k; %square mean root of sq4 of month m
```

```
%%% T850 PCs/Centroids of PCs belonging to the same Z500 cluster/...
% Z500 centroids/PCs same as Z500 representatives/...
% PCs same as Z500 Best total representative %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    sq1=zeros(k,1);
    sq2=zeros(k,1);
    sq3=zeros(k,1);
    sq4=zeros(k,1);
    for j=1:eof_modes
        sq1=(C_T850(:,j)-C(:,j)).^2+sq1;
        sq2=(C_T850(:,j)-pc_T850_norm(j,crti)').^2+sq2;
        sq3=(C(:,j)-pc_T850_norm(j,crti)').^2+sq3;
        sq4=(C_T850(:,j)-pc_T850_norm(j,crti1)').^2+sq4;
    end

    er1_T850(m)=sum(sqrt(sq1))/k;
    er2_T850(m)=sum(sqrt(sq2))/k;
    er3_T850(m)=sum(sqrt(sq3))/k;
    er4_T850(m)=sum(sqrt(sq4))/k;

%%% SLHF PCs/Centroids of PCs belonging to the same Z500 cluster/...
                        %Z500 centroids/PCs same as Z500 representatives
    sq1=zeros(k,1);
    sq2=zeros(k,1);
    sq3=zeros(k,1);
    sq4=zeros(k,1);
    for j=1:eof_modes
        sq1=(C_SLHF(:,j)-C(:,j)).^2+sq1;
        sq2=(C_SLHF(:,j)-pc_SLHF_norm(j,crti)').^2+sq2;
        sq3=(C(:,j)-pc_SLHF_norm(j,crti)').^2+sq3;
        sq4=(C_SLHF(:,j)-pc_SLHF_norm(j,crti1)').^2+sq4;
    end

    er1_SLHF(m)=sum(sqrt(sq1))/k;
    er2_SLHF(m)=sum(sqrt(sq2))/k;
    er3_SLHF(m)=sum(sqrt(sq3))/k;
    er4_SLHF(m)=sum(sqrt(sq4))/k;

%%% W300 PCs/Centroids of PCs belonging to the same Z500 cluster/...
                        %Z500 centroids/PCs same as Z500 representatives

    sq1=zeros(k,1);
    sq2=zeros(k,1);
    sq3=zeros(k,1);
    sq4=zeros(k,1);
    for j=1:eof_modes
        sq1=(C_W300(:,j)-C(:,j)).^2+sq1;
        sq2=(C_W300(:,j)-pc_W300_norm(j,crti)').^2+sq2;
        sq3=(C(:,j)-pc_W300_norm(j,crti)').^2+sq3;
        sq4=(C_W300(:,j)-pc_W300_norm(j,crti1)').^2+sq4;
    end

    er1_W300(m)=sum(sqrt(sq1))/k;
    er2_W300(m)=sum(sqrt(sq2))/k;
    er3_W300(m)=sum(sqrt(sq3))/k;
    er4_W300(m)=sum(sqrt(sq4))/k;

%%% PV850 PCs/Centroids of PCs belonging to the same Z500 cluster/...
                        %Z500 centroids/PCs same as Z500 representatives
    sq1=zeros(k,1);
    sq2=zeros(k,1);
    sq3=zeros(k,1);
    sq4=zeros(k,1);
    for j=1:eof_modes
        sq1=(C_PV850(:,j)-C(:,j)).^2+sq1;
        sq2=(C_PV850(:,j)-pc_PV850_norm(j,crti)').^2+sq2;
        sq3=(C(:,j)-pc_PV850_norm(j,crti)').^2+sq3;
        sq4=(C_PV850(:,j)-pc_PV850_norm(j,crti1)').^2+sq4;
    end

    er1_PV850(m)=sum(sqrt(sq1))/k;
    er2_PV850(m)=sum(sqrt(sq2))/k;
    er3_PV850(m)=sum(sqrt(sq3))/k;
    er4_PV850(m)=sum(sqrt(sq4))/k;

%%% W850 PCs/Centroids of PCs belonging to the same Z500 cluster/...
                        %Z500 centroids/PCs same as Z500 representatives
    sq1=zeros(k,1);
```

```
    sq2=zeros(k,1);
    sq3=zeros(k,1);
    sq4=zeros(k,1);
    for j=1:eof_modes
        sq1=(C_W850(:,j)-C(:,j)).^2+sq1;
        sq2=(C_W850(:,j)-pc_W850_norm(j,crti)').^2+sq2;
        sq3=(C(:,j)-pc_W850_norm(j,crti)').^2+sq3;
        sq4=(C_W850(:,j)-pc_W850_norm(j,crti1)').^2+sq4;
    end

    er1_W850(m)=sum(sqrt(sq1))/k;
    er2_W850(m)=sum(sqrt(sq2))/k;
    er3_W850(m)=sum(sqrt(sq3))/k;
    er4_W850(m)=sum(sqrt(sq4))/k;

%%% SSHF PCs/Centroids of PCs belonging to the same Z500 cluster/...
                        %Z500 centroids/PCs same as Z500 representatives
    sq1=zeros(k,1);
    sq2=zeros(k,1);
    sq3=zeros(k,1);
    sq4=zeros(k,1);
    for j=1:eof_modes
        sq1=(C_SSHF(:,j)-C(:,j)).^2+sq1;
        sq2=(C_SSHF(:,j)-pc_SSHF_norm(j,crti)').^2+sq2;
        sq3=(C(:,j)-pc_SSHF_norm(j,crti)').^2+sq3;
        sq4=(C_SSHF(:,j)-pc_SSHF_norm(j,crti1)').^2+sq4;
    end

    er1_SSHF(m)=sum(sqrt(sq1))/k;
    er2_SSHF(m)=sum(sqrt(sq2))/k;
    er3_SSHF(m)=sum(sqrt(sq3))/k;
    er4_SSHF(m)=sum(sqrt(sq4))/k;


%%% PV315 PCs/Centroids of PCs belonging to the same Z500 cluster/...
                        %Z500 centroids/PCs same as Z500 representatives
    sq1=zeros(k,1);
    sq2=zeros(k,1);
    sq3=zeros(k,1);
    sq4=zeros(k,1);
    for j=1:eof_modes
        sq1=(C_PV315(:,j)-C(:,j)).^2+sq1;
        sq2=(C_PV315(:,j)-pc_PV315_norm(j,crti)').^2+sq2;
        sq3=(C(:,j)-pc_PV315_norm(j,crti)').^2+sq3;
        sq4=(C_PV315(:,j)-pc_PV315_norm(j,crti1)').^2+sq4;
    end

    er1_PV315(m)=sum(sqrt(sq1))/k;
    er2_PV315(m)=sum(sqrt(sq2))/k;
    er3_PV315(m)=sum(sqrt(sq3))/k;
    er4_PV315(m)=sum(sqrt(sq4))/k;

%%% PT2PVU PCs/Centroids of PCs belonging to the same Z500 cluster/...
                        %Z500 centroids/PCs same as Z500 representatives
    sq1=zeros(k,1);
    sq2=zeros(k,1);
    sq3=zeros(k,1);
    sq4=zeros(k,1);
    for j=1:eof_modes
        sq1=(C_PT2PVU(:,j)-C(:,j)).^2+sq1;
        sq2=(C_PT2PVU(:,j)-pc_PT2PVU_norm(j,crti)').^2+sq2;
        sq3=(C(:,j)-pc_PT2PVU_norm(j,crti)').^2+sq3;
        sq4=(C_PT2PVU(:,j)-pc_PT2PVU_norm(j,crti1)').^2+sq4;
    end

    er1_PT2PVU(m)=sum(sqrt(sq1))/k;
    er2_PT2PVU(m)=sum(sqrt(sq2))/k;
    er3_PT2PVU(m)=sum(sqrt(sq3))/k;
    er4_PT2PVU(m)=sum(sqrt(sq4))/k;

end

figure(count);
m=1:12;
plot(m,(er1_MSLP+er1_T850+er1_SLHF+er1_W300+er1_PV850+er1_W850+er1_SSHF+er1_PV315+er1_PT2PVU)/
9)
hold on
```

```
plot(m,(er2_MSLP+er2_T850+er2_SLHF+er2_W300+er2_PV850+er2_W850+er2_SSHF+er2_PV315+er2_PT2PVU)/
9)
hold on
plot(m,(er3_Z500+er3_MSLP+er3_T850+er3_SLHF+er3_W300+er3_PV850+er3_W850+er3_SSHF+er3_PV315+er3
_PT2PVU)/10)
hold on
plot(m,(er4_Z500+er4_MSLP+er4_T850+er4_SLHF+er4_W300+er4_PV850+er4_W850+er4_SSHF+er4_PV315+er4
_PT2PVU)/10)
hold off
legend({'MEAN ERROR: Parameters Centroid-Z500 Centroid','MEAN ERROR: Parameters Centroid-Z500
Representative','MEAN ERROR: Z500 centroid-Z500 representatives','MEAN ERROR: Parameters
Centroid-Z500 Best Representative'},'Location','best')
xlabel('Months')
ylabel('Square mean root distance')
xlim([1 12])
ylim([0 inf])
title(['Mean Total Error: Centroids Vs Representatives for EOF modes =',num2str(eof_modes)])
f=gcf;
set(f, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1]); %full screen

count=count+1;
w = waitforbuttonpress; %comment out if you want to change frame manually
end
```

Functions for plotting each parameters fields

Below are the functions for plotting the fields of each meteorological parameter:

```
function PLOT_MSLP(longitude,latitude,MSLP_2D,average_window)
%Plots the MSLP isolines
%longitude: the longitude vector
%latitude: the latitude vector
%MSLP_2D : the Lat x Lon matrix (screenshot) we want to plot
%average_window (moving average window):
%        if ==0, no average is applied
%        if ==n, (n: odd integer) it averages 'n' points in both directions

%plot Coastlines
Coast = load('coast.mat');
plot(Coast.long,Coast.lat,'k')
axis equal
xlim([min(longitude) max(longitude)])
ylim([min(latitude) max(latitude)])
hold on

%plot MSLP isolines
clow=988; chigh=1048; cint=2;      % contour line limits
conts = [clow:cint*2:chigh];       % every 2 line
V = [clow+cint:cint*2:chigh];      % every 2 line between the obove lines

if average_window==0
    [c1,h1]=contour(longitude,latitude,MSLP_2D,conts,'black'); %raw data
    hold on
    contour(longitude,latitude,MSLP_2D,V,'black','LineWidth',1);
else %apply moving average on both dimensions to eliminate noise
    window=average_window; %moving average window
    mslp_2D_movmean= movmean(MSLP_2D,window,1);
    mslp_2D_movmean= movmean(mslp_2D_movmean,window,2);
    [c1,h1]=contour(longitude,latitude,mslp_2D_movmean,conts,'black'); %moving mean result
    hold on
    contour(longitude,latitude,mslp_2D_movmean,V,'black','LineWidth',1);
end

h1.LineWidth = 2;
clabel(c1,h1,'LabelSpacing',800,'Color','black') %puts in contour values
hold off

end
```

```
function PLOT_PT2PVU(longitude,latitude,PT2PVU_2D,average_window)
%Plot 2PVU Potential Temperature Pcolor
```

```
% longitude :longitude vector
% latitude : latitude vector
% PT2PVU_2D : 2D matrix Lat x Lon (screenshot) we want to plot
%average_window (moving average window):
%        if ==0, no average is applied
%        if ==n, (n: odd integer) it averages 'n' points in both directions

if average_window==0
    %do nothing (no moving average)
else %apply moving average on both dimensions to eliminate noise
    window=average_window; %moving average window
    PT2PVU_2D_movmean= movmean(PT2PVU_2D,window,1);
    PT2PVU_2D_movmean= movmean(PT2PVU_2D_movmean,window,2);
    PT2PVU_2D=PT2PVU_2D_movmean;
end

%Plot pcolor
minc=300;   maxc=380;   intc=8;
number_of_shades=(maxc-minc)/intc;
colormap(jet(number_of_shades))

pcolor(longitude, latitude, PT2PVU_2D); shading interp;
caxis([minc maxc]) % pcolor data range
c=colorbar;
c.Ticks = linspace(minc,maxc,number_of_shades+1); % tick and label
c.TickLabels = string(linspace(minc,maxc,number_of_shades+1));
c.Label.String = 'Kelvin (K)';
hold on

%Plot coastlines
Coast = load('coast.mat');
plot(Coast.long,Coast.lat,'k')
axis equal
xlim([min(longitude) max(longitude)])
ylim([min(latitude) max(latitude)])
hold on

%Plot contours
conts = [minc:intc:maxc];        % define contour lines
[c,h]=contour(longitude,latitude,PT2PVU_2D,conts,'black'); %non interpolated result
h.LineWidth = 0.1;
clabel(c,h,'Color','black')      %puts in contour value
hold off

end
```

```
function PLOT_PV315(longitude,latitude,PV315_2D,average_window)
%Plots the 315K Potential Vorticity Isolines
%longitude: the longitude vector
%latitude: the latitude vector
%PV315_2D : the Lat x Lon matrix (screenshot) we want to plot
%average window (moving average window):
%        if ==0, no average is applied
%        if ==n, (n: odd integer) it averages 'n' points in both directions

%plot Coastlines
Coast = load('coast.mat');
plot(Coast.long,Coast.lat,'k')
axis equal
xlim([min(longitude) max(longitude)])
ylim([min(latitude) max(latitude)])
hold on

%plot PV315 isolines
clow=-2; czero=0; chigh=10; cint=1;     % contour line limits
conts  = [clow:cint:chigh];             % define contour lines all
conts1 = [clow:cint:czero-cint];        % define contour lines negative
conts2 = [czero+cint:cint:chigh];       % define contour lines positive

if average_window==0 %raw data
    [c,h]=contour(longitude,latitude,PV315_2D,conts1,'magenta');
    [c1,h1]=contour(longitude,latitude,PV315_2D,conts2,'magenta');
else %apply moving average on both dimensions to eliminate noise
    window=average_window;
    PV315_movmean= movmean(PV315_2D,window,1);
```

```
    PV315_movmean= movmean(PV315_movmean,window,2);
    [c,h]=contour(longitude,latitude,PV315_movmean,conts1,'magenta');       %moving mean result
positive
    [c1,h1]=contour(longitude,latitude,PV315_movmean,conts2,'magenta');     %moving mean result
negative
end

h.LineWidth = 2;
clabel(c,h,'Color','magenta')       %puts in contour labels positive
h1.LineWidth = 2;
clabel(c1,h1,'Color','magenta')       %puts in contour labels negative
hold off

end
```

```
function PLOT_PV850(longitude,latitude,PV850_2D)
%Plots the 850hPa Potential Vorticity Pcolor
%longitude: the longitude vector
%latitude: the latitude vector
%PV850_2D : the Lat x Lon matrix (screenshot) we want to plot

minc=-2;    maxc=38;    number_of_shades=10;
pcolor(longitude, latitude, PV850_2D); shading interp;
a=flipud(hot(number_of_shades+4));%more shades
b=a([1 5:13],:);% remove the yellow shades
colormap(b)%custom colormap
caxis([minc maxc]) % pcolor data range
c=colorbar;
c.Ticks = linspace(minc,maxc,number_of_shades+1); % tick and label skipping one level
c.TickLabels = string(linspace(minc,maxc,number_of_shades+1));
c.Label.String = 'PVU';
hold on

%Plot Coastlines
Coast = load('coast.mat');
plot(Coast.long,Coast.lat,'k')
axis equal
xlim([min(longitude) max(longitude)])
ylim([min(latitude) max(latitude)])
hold off

end
```

```
function PLOT_SLHF(longitude, latitude,SLHF_2D,option)
%Plots the Surface Latent Heat Flux Pcolor
%longitude: the longitude vector
%latitude: the latitude vector
%SLHF_2D : the Lat x Lon matrix (screenshot) we want to plot
%option  ==1 : full range, includes positive heat flux (downwards heat)
%        ==2 : only negative heat flux

if option==1
    minc=-400; maxc=25;
elseif option==2
    minc=-400; maxc=0;
end
colorstep=25;
range=abs(minc)+abs(maxc);
colors=range/colorstep;
colormap(hot(colors))

%Plot pcolor
pcolor(longitude, latitude, SLHF_2D); shading interp;
caxis([minc maxc]) % pcolor data range
c=colorbar;
c.Label.String = 'W/m^2';
hold on

%%% Plot coastlines
Coast = load('coast.mat');
plot(Coast.long,Coast.lat,'k')
axis equal
xlim([min(longitude) max(longitude)])
```

```
ylim([min(latitude) max(latitude)])
hold off
end
```

```
function PLOT_SSHF(longitude, latitude, SSHF_2D,option)
%Plots the Surface Sensible Heat Flux Pcolor
%longitude: the longitude vector
%latitude: the latitude vector
%SSHF_2D : the Lat x Lon matrix (screenshot) we want to plot
%option  ==1 : full range, includes positive heat flux (downwards heat)
%        ==2 : only negative heat flux

%Plot pcolor
colorstep=25;

if option==1
    minc=-225; maxc=50;
    range=abs(minc)+abs(maxc);
    colors=range/colorstep;
    colormap(hot(colors))
elseif option==2
    minc=-225; maxc=0; %only negative heat flux
    range=abs(minc)+abs(maxc);
    colors=range/colorstep;
    a=hot(colors+4);
    b=a(1:colors,:);
    b(colors,:)=[1 1 1];
    colormap(b)
end

pcolor(longitude, latitude, SSHF_2D); shading interp;
caxis([minc maxc]) % pcolor data range
c=colorbar;
c.Label.String = 'W/m^2';
hold on

%Plot coastlines
Coast = load('coast.mat');
plot(Coast.long,Coast.lat,'k')
axis equal
xlim([min(longitude) max(longitude)])
ylim([min(latitude) max(latitude)])
hold off

end
```

```
function PLOT_T850(longitude,latitude,T850_2D)
%Plots the 850hPa Temperature Pcolor
%longitude: the longitude vector
%latitude: the latitude vector
%T850_2D : the Lat x Lon matrix (screenshot) we want to plot

%plot T850 pcolor
pcolor(longitude, latitude, T850_2D); shading interp;
colormap(jet(12))
caxis([-20 40]) % pcolor data range
c=colorbar;
c.Label.String = 'Celsius';
hold on

%plot Coastlines
Coast = load('coast.mat');
plot(Coast.long,Coast.lat,'k')
axis equal
xlim([min(longitude) max(longitude)])
ylim([min(latitude) max(latitude)])
hold on

clow=-20; chigh=40; cint=5;   % contour line limits
conts = [clow:cint:chigh];        % define contour lines
V = [clow:cint*2:chigh];          % label every 2th line
[c,h]=contour(longitude,latitude,T850_2D,conts,'black'); %non interpolated result
h.LineWidth = 0.1;
```

```
clabel(c,h,'Color','black')      %puts in contour value
%clabel(c,h,V)                   %puts in contours
hold off

end
```

```
function PLOT_W300(longitude,latitude,U300_3D,V300_3D,U300_2D,V300_2D,W300_2D,scale_mode)
%Plots the 300hPa winds
%longitude: the longitude vector
%latitude: the latitude vector
%U300_3D : the 3D matrix containing all the U850 values. In order to
%extract the max Value (for scaling)
%V300_3D : the same as above for V850
%V300_2D : the Lat x Lon matrix (screenshot) we want to plot
%V300_2D : the Lat x Lon matrix (screenshot) we want to plot
%scale_mode  ==1 : build in automatic scaling of arrows in quiver function
%            ==2 : custom scaling based on Max.U850 or Max.V850
%UNTITLED3 Summary of this function goes here
%   Detailed explanation goes here
%plot W300 pcolor %%%%%%%%%%%%%%%%%%%%%%%%%%
pcolor(longitude, latitude, W300_2D); shading interp;
number_of_shades=9;
a=flipud(parula(number_of_shades));
a(1,:)=[1 1 1];
colormap(a)
minc=30;
maxc=120;
caxis([minc maxc]) % pcolor data range
c=colorbar;
c.Ticks = linspace(minc,maxc,number_of_shades+1); % tick and label skipping one level
c.TickLabels = string(linspace(minc,maxc,number_of_shades+1));
c.Label.String = 'Knots';
hold on

%plot coastlines
Coast = load('coast.mat');
plot(Coast.long,Coast.lat,'k')
axis equal
xlim([min(longitude) max(longitude)])
ylim([min(latitude) max(latitude)])
hold on

%apply interpolation to show less data (very small arrows)
newpointsx = 20;
newpointsy = 8;
[xq,yq] = meshgrid(...
  linspace(min(min(longitude,[],2)),max(max(longitude,[],2)),newpointsx ),...
  linspace(min(min(latitude,[],1)),max(max(latitude,[],1)),newpointsy ));
U300_2D_smooth = interp2(longitude,latitude,U300_2D,xq,yq,'cubic');
V300_2D_smooth = interp2(longitude,latitude,V300_2D,xq,yq,'cubic');

%plot quiver 2D arrows
if scale_mode==1 %build in quiver autoscale on
    quiver(xq,yq,U300_2D_smooth,V300_2D_smooth,'black') %automatic
%or
elseif scale_mode==2 %custom scaling uniform for all plots
    stepx=(max(xq(:))-min(xq(:)))/(newpointsx-1);
    stepy=(max(yq(:))-min(yq(:)))/(newpointsy-1);
    scale=min(stepx/max(U300_3D(:)),stepy/max(V300_3D(:))); %Global scale...
                                    %factor so that arrows do not overlap
    quiver(xq,yq,U300_2D_smooth.*scale,V300_2D_smooth.*scale,0,'black') %autoscale off
end

hold off
end
```

```
function PLOT_W850(longitude,latitude,U850_3D,V850_3D,U850_2D,V850_2D,W850_2D,scale_mode)
%Plots the 850hPa winds
%longitude: the longitude vector
%latitude: the latitude vector
%U850_3D : the 3D matrix containing all the U850 values. In order to
%extract the max Value (for scaling)
%V850_3D : the same as above for V850
```

```
%V850_2D : the Lat x Lon matrix (screenshot) we want to plot
%V850_2D : the Lat x Lon matrix (screenshot) we want to plot
%scale_mode  ==1 : build in automatic scaling of arrows in quiver function
%            ==2 : custom scaling based on Max.U850 or Max.V850

%plot 850hPa winds pcolor
pcolor(longitude, latitude, W850_2D); shading interp;
number_of_shades=6;
a=flipud(hot(number_of_shades+2));
b=a([1 3:7],:);
colormap(b)
%colormap(flipud(hot(number_of_shades)))
minc=10;
maxc=40;
caxis([minc maxc]) % pcolor data range
c=colorbar;
c.Ticks = linspace(minc,maxc,number_of_shades+1); % tick and label skipping one level
c.TickLabels = string(linspace(minc,maxc,number_of_shades+1));
c.Label.String = 'Knots';
hold on

%plot coastlines
Coast = load('coast.mat');
plot(Coast.long,Coast.lat,'k')
axis equal
xlim([min(longitude) max(longitude)])
ylim([min(latitude) max(latitude)])
hold on

%apply interpolation to show less data (very small arrows)
newpointsx = 20;
newpointsy = 8;
[xq,yq] = meshgrid(...
    linspace(min(min(longitude,[],2)),max(max(longitude,[],2)),newpointsx ),...
    linspace(min(min(latitude,[],1)),max(max(latitude,[],1)),newpointsy ));
U850_2D_smooth = interp2(longitude,latitude,U850_2D,xq,yq,'cubic');
V850_2D_smooth = interp2(longitude,latitude,V850_2D,xq,yq,'cubic');

%plot quiver 2D arrows
if scale_mode==1 %build in quiver autoscale on
    quiver(xq,yq,U850_2D_smooth,V850_2D_smooth,'black') %automatic
%or
elseif scale_mode==2 %custom scaling uniform for all plots
    stepx=(max(xq(:))-min(xq(:)))/(newpointsx-1);
    stepy=(max(yq(:))-min(yq(:)))/(newpointsy-1);
    scale=min(stepx/max(U850_3D(:)),stepy/max(V850_3D(:))); %Global scale...
                                    %factor so that arrows do not overlap
    quiver(xq,yq,U850_2D_smooth.*scale,V850_2D_smooth.*scale,0,'black') %autoscale off
end

hold off
end
```

```
function PLOT_Z500(longitude,latitude,Z500_2D)
%Plots the 500hPa Geopotential Height Isolines
%longitude: the longitude vector
%latitude: the latitude vector
%Z500_2D : the Lat x Lon matrix (screenshot) we want to plot

%plot coastlines %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Coast = load('coast.mat');
plot(Coast.long,Coast.lat,'k')
axis equal
xlim([min(longitude) max(longitude)])
ylim([min(latitude) max(latitude)])
hold on

clow=4680; chigh=6000; cint=60;   % contour line limits
conts = [clow:cint:chigh];        % define contour lines
[c,h]=contour(longitude,latitude,Z500_2D,conts,'blue');
h.LineWidth = 2;
clabel(c,h,'LabelSpacing',400,'Color','blue') %display contour labels
%clabel(c,h,V)                  %display contour labels skipping one
hold off
end
```

Appendix 5: [MATLAB script for SOM]

SOM clustering

Based on section 1.2.5, using MATLAB's "Statistics and Machine Learning Toolbox"[13].

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SOM Clustering of the EOF data & of Full Patterns
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    ci=zeros(length(t2),1);     %Vector with cluster indices

% loop for each month %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for m=min(month2):max(month2)
    close all

    %if we want to split all EOF (ex Januaries) into k clusters:
    index= find(month2==m);        %January position index in t2 array
    pc_month=pc_Z500_norm(1:eof_modes,index)'; %Principle Components containing only...
                                   %the Januaries, Transpose (: columns are...
                                   %the variables and rows are the points)

    net = selforgmap([2 2]); % determine the neurons
    net = train(net,pc_month'); % train the algorithm
    D=net.IW{1}; % returns the neuron weights (centroids)

    %%% Cluster Reference Full Pattern %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %Find the full pattern time step closest to the centroids.
    %Use of Root Mean Square Distance (RMSD) to determine min distance from...
    %centroid

    RMSD=zeros(size(pc_month,1),k); %matrix containing all the L2 distances...
                                    %of the PCs from the centroids
                                    %-rows: clusters
                                    %-columns: RMSD of PCs from cluster centroid
    for i=1:k
    sq_error=zeros(size(pc_month,1),1); %empty column vector with the same...
                                        %length as pc_month
        for j=1:eof_modes
            sq_error=(pc_month(:,j)-D(i,j)).^2+sq_error; %sum of L2 distance of...
                                                         %all PC elements from cluster #i centroid
        end
    %RMSD(:,i)=sqrt(sq_error./eof_modes);
    RMSD(:,i)=sqrt(sq_error);
    end

    %%% Cluster Reference Full Pattern %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %Find the full pattern time step closest to the centroids.

    [M,I]=min(RMSD);    %M:the min of each column in cluster i
                        %I:index of the first occurrence of min (in case...
                        %  multiple distances of columns of D (same cluster)...
                        %  are equal, keep only the fist occurrence and its index
    %so the most representative full Z500 patterns are for example:
                         %- the 15th January from the 41(years) in total
                         %- the 39th January from the 41 in total
                         %- the 16th January from the 41 in total
                         %- the  1st January from the 41 in total
    %each belonging to deferent cluster

    crti=index(I);       % cluster reference time indices in t2 array
    ci(index)=idx;       % update the cluster indices (ci) vector
    datestr(t2(crti))    % to see the dates of min Distance from centroid

end
```

[13] https://www.mathworks.com/products/statistics.html
https://www.mathworks.com/help/deeplearning/gs/cluster-data-with-a-self-organizing-map.html