ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

ΤΟΜΕΑΣ ΘΕΡΜΟΤΗΤΑΣ

Εργαστήριο Θερμικών Διεργασιών

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΘΕΡΜΟΔΥΝΑΜΙΚΗ ΑΝΑΛΥΣΗ ΤΟΥ ΤΡΙΜΕΡΟΥΣ ΜΙΓΜΑΤΟΣ R152A/R125/R32 ΜΕ ΤΗΝ ΚΑΤΑΣΤΑΤΙΚΗ ΕΞΙΣΩΣΗ PENG-ROBINSON

της προπτυχιακής φοιτήτριας

ΠΑΠΠΑ ΕΥΑΣ

Επιβλέπουσα Καθηγήτρια:

Στέγγου-Σαγιά Αθηνά

Αθήνα 2010

Περίληψη

Σε αυτή τη διπλωματική εργασία δοκιμάστηκε το μοντέλο καταστατικής εξίσωσης Peng-Robinson για τα υπέρθερμα αέρια μίγματα R407b και R152a / R125 / R32 με κατά μάζα σύσταση 48 / 18 / 34. Το δεύτερο είναι ένα νέο μίγμα με πολύ μικρή επίδραση στο περιβάλλον σε σχέση με τα σημερινά ψυκτικά μέσα και από πρόσφατες μελέτες φαίνεται ικανό να αντικαταστήσει το R22.

Με σκοπό τον προσδιορισμό των θερμοφυσικών ιδιοτήτων του νέου αερίου στην περιοχή λειτουργίας δημιουργήθηκε υπολογιστικός κώδικας στη γλώσσα προγραμματισμού java. Επιλέχθηκε να μελετηθούν η πίεση, η εντροπία, η ενθαλπία, η ειδική θερμοχωρητικότητα υπό σταθερό όγκο, η ειδική θερμοχωρητικότητα υπό σταθερή πίεση, ο λόγος ειδικών θερμοχωρητικοτήτων, οι ισεντροπικοί συντελεστές, ο συντελεστής συμπιεστότητας και η ταχύτητα ήχου. Το εύρος θερμοκρασίας ορίστηκε 240,15 K - 481,05 K και το εύρος όγκου 0,006 m^3/kg – 0,6 m^3/kg. Για κάθε μέγεθος εκπονήθηκε διάγραμμα συναρτήσει της θερμοκρασίας και της πίεσης.

Στη συνέχεια πραγματοποιήθηκε συγκριτική ανάλυση των αποτελεσμάτων με στοιχεία που αντλήθηκαν από το πρόγραμμα REFPROP της NIST και υπολογίστηκαν τα σχετικά σφάλματα. Για τον εύκολο προσδιορισμό των σφαλμάτων δημιουργήθηκαν τα αντίστοιχα διαγράμματα των ιδιοτήτων που παρουσιάζουν μεγάλες αποκλίσεις.

Η εργασία καταλήγει με το συμπέρασμα ότι το μοντέλο της Peng-Robinson περιγράφει ικανοποιητικά όλα τα υπό εξέταση μεγέθη εκτός από τις ειδικές θερμοχωρητικότητες υπό σταθερό όγκο και σταθερή πίεση. Γι’ αυτές τις δύο μεταβλητές εμφανίζονται υψηλές αποκλίσεις για μεγάλη περιοχή θερμοκρασιών και όγκων, οι οποίες καθιστούν ακατάλληλη τη χρήση του μοντέλου αυτού.

Abstract

This thesis applies the Peng-Robinson equation of state (EoS) for the superheated gas mixture R407b and R152a / R125 / R32 with a mass ratio 48 / 18 / 34. The last mentioned gas is a new blend with little impact on the environment in comparison to current refrigerants and from recent studies it seems to have an ideal potential to be a drop-in replacement of R22.

With the aim to investigate the thermophysical characteristics of this new mixture in the operating region, a computer code has been developed with the java programming language. The selected properties were pressure, entropy, enthalpy, specific heat capacity at constant volume, specific heat capacity at constant pressure, ratio of specific heat capacities, compressibility factor and speed of sound. The range of temperature was set from 240,15 K to 481,05 K and the range of volume from 0,006 m^3/kg to 0,6 m^3/kg. The diagrams of every variable were produced as a function of temperature and pressure.

Afterwards, a comparative analysis was performed using the REFPROP program of NIST to estimate the relative errors of the calculations. For the easy determination of the errors the corresponding diagrams of the properties with high deviations were created and presented.

Finally, this work concludes that the Peng-Robinson EoS simulates with adequacy the performance of all characteristics studied except for the specific heat capacities. For these two variables high deviations were measured in a large range of temperature and volume, which indicates that this type of equation of state is unsuitable.

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω την επιβλέπουσα καθηγήτρια της διπλωματικής μου εργασίας κ. Στέγγου-Σαγιά Αθηνά για την εμπιστοσύνη της και τη δυνατότητα που μου έδωσε να ασχοληθώ με το αντικείμενο των φιλικών προς το περιβάλλον ψυκτικών μέσων. Η εργασία αυτή δε θα μπορούσε να είχε πραγματωθεί χωρίς την προσωπική καθοδήγηση και τις συμβουλές της.

Ιδιαίτερα θα ήθελα να ευχαριστήσω τον αδερφό μου Παππά Χρίστο, στον οποίο οφείλω τις προγραμματιστικές γνώσεις μου, για την υπομονή και τη διάθεση που επέδειξε.

Θα ήθελα ακόμα να αναφέρω όλους εκείνους που έμμεσα ή άμεσα μου συμπαραστάθηκαν στην εκπόνηση της εργασίας αυτής, φίλους και συναδέλφους καθώς και την κ. Καίτη.

Κυρίως όμως οφείλω να ευχαριστήσω την οικογένειά μου για τη στήριξη που μου παρείχε κατά τη διάρκεια των σπουδών μου, ηθική και οικονομική.

Περιεχόμενα

Γενικά

Από τη δεκαετία του ‘80 οπότε παρατηρήθηκε η ζημιά που επιφέρουν τα ψυκτικά μέσα στη στιβάδα του όζοντος, γίνονται συνεχείς προσπάθειες για την ανεύρεση φιλικότερων προς το περιβάλλον εργαζόμενων μέσων. Μέχρι και σήμερα το πρόβλημα δεν έχει λυθεί. Αν και οι βλαβερές ουσίες για το όζον έχουν απομακρυνθεί από την αγορά, τα χρησιμοποιούμενα ψυκτικά έχουν πολλαπλάσιο αντίκτυπο στο φαινόμενο του θερμοκηπίου από το διοξείδιο του άνθρακα.

Τα χαρακτηριστικά και οι ιδιότητες των υποψήφιων ψυκτικών είναι το αντικείμενο μελέτης των επιστημόνων που αναζητούν πιο «πράσινες» λύσεις. Σε αυτό το πλαίσιο που ορίζει η παγκόσμια κατάσταση, δραστηριοποιείται και το εργαστήριο θερμικών διεργασιών της σχολής των Μηχανολόγων του ΕΜΠ. Η παρούσα εργασία δημιουργήθηκε με σκοπό την εκπόνηση υπολογιστικού κώδικα, ο οποίος να μπορεί να περιγράψει τη συμπεριφορά ενός νέου ψυκτικού μίγματος (R152a/R125/R32) στην υπέρθερμη κατάσταση. Το νέο αυτό μίγμα φαίνεται ηπιότερο ως προς την επίδραση του στο περιβάλλον και συγκεντρώνει πολλά χαρακτηριστικά, τα οποία θα μπορούσαν να το χρίσουν μελλοντικά αντικαταστάτη του R22 [1].

Η διαδικασία που ακολουθήθηκε για τη δημιουργία του κώδικα, την εύρεση των θερμοφυσικών ιδιοτήτων και τον προσδιορισμό των αποκλίσεων των υπολογισμών περιγράφεται στα ακόλουθα κεφάλαια. Σε αυτό το σημείο δίνεται η διάρθρωσή τους στο σύνολο της εργασίας.

Στο πρώτο κεφάλαιο γίνεται μια εισαγωγή, μια σύντομη ιστορική αναδρομή και μια επεξήγηση των λόγων που οδήγησαν στην επιλογή των δύο μιγμάτων που τελικά μελετούνται στη διπλωματική εργασία.

Στο δεύτερο κεφάλαιο αρχικά παρουσιάζεται η καταστατική εξίσωση (Equation of State - EoS) που χρησιμοποιήθηκε ως μοντέλο προσομοίωσης στην αέρια φάση. Στη συνέχεια αναλύεται το μαθηματικό υπόβαθρο και το σύνολο των σχέσεων που επέτρεψαν τον υπολογισμό των επιθυμητών μεγεθών, ενώ στο τελευταίο μέρος περιγράφεται η δομή και τα χαρακτηριστικά του κώδικα που υλοποιήθηκε.

Στο τρίτο κεφάλαιο αναφέρονται όλα τα απαραίτητα στοιχεία και σταθερές των ουσιών που απαιτήθηκαν ως δεδομένα εισαγωγής στο πρόγραμμα το οποίο κατασκευάστηκε.

Στο τελευταίο και τέταρτο κεφάλαιο παρουσιάζονται τα αποτελέσματα της μελέτης, τα σφάλματα που εμφανίστηκαν καθώς και ο σχολιασμός και η αξιολόγησή τους.

Η εργασία ολοκληρώνεται με την παράθεση τριών παραρτημάτων. Το παράρτημα Α περιλαμβάνει τα διαγράμματα όλων των θερμοδυναμικών μεγεθών που υπολογίστηκαν συναρτήσει της θερμοκρασίας και της πίεσης. Στο παράρτημα Β φαίνονται τα διαγράμματα σφαλμάτων που κρίθηκε χρήσιμο να παρουσιαστούν σε συνάρτηση με τη θερμοκρασία και τον όγκο. Τέλος, στο παράρτημα Γ δίνεται ο κώδικας που δημιουργήθηκε με παράλληλη επεξήγηση κάθε βήματος.

Κεφάλαιο 1° :

1.1 Εισαγωγή

Στην παρούσα διπλωματική εργασία παρουσιάζεται η μελέτη των θερμοφυσικών ιδιοτήτων του ψυκτικού μέσου R407b και του μίγματος R152a/R125/R32 σε αναλογίες 48/18/34 κατά μάζα στην υπέρθερμη περιοχή.

Η εύρεση των θερμοδυναμικών μεγεθών σε αυτή την περιοχή προαπαιτεί τη γνώση δύο ανεξάρτητων μεταβλητών, δηλαδή το σύστημα έχει δύο βαθμούς ελευθερίας. Θεωρώντας ως ανεξάρτητες μεταβλητές τη θερμοκρασία και τον όγκο, η περιγραφή του αερίου καθορίζεται από την επιλογή της καταστατικής εξίσωσης. Σε αυτήν την εργασία δοκιμάστηκε το μοντέλο της Peng-Robinson και με χρήση των μαθηματικών και της θερμοδυναμικής θεωρίας καταστρώθηκαν όλες οι απαιτούμενες εξισώσεις για την εξεύρεση των επιθυμητών μεγεθών. Ως τέτοια επιλέχθηκαν η πίεση, η εντροπία, η ενθαλπία, η ειδική θερμοχωρητικότητα υπό σταθερό όγκο, η ειδική θερμοχωρητικότητα υπό σταθερή πίεση, οι ισεντροπικοί συντελεστές, ο λόγος των ειδικών θερμοχωρητικοτήτων, ο συντελεστής συμπιεστότητας και η ταχύτητα του ήχου.

Για τον υπολογισμό τους στο επιθυμητό εύρος θερμοκρασίας και όγκου διαμορφώθηκε κατάλληλος κώδικας στη γλώσσα προγραμματισμού java. Επειδή και τα δύο μίγματα βρίσκουν εφαρμογή ως αντικαταστάτες του R22, οι περιοχές θερμοκρασίας και όγκου που μελετήθηκαν είναι κοντά στις τιμές λειτουργίας αυτού. Με αυτό το σκεπτικό ως εύρος θερμοκρασίας ορίστηκε το 240,15K – 481,05K και όγκου το $0,006 m^3/kg$ – $0,6 m^3/kg$.

Τα εξαγόμενα αποτελέσματα του προγράμματος χρησιμοποιήθηκαν για τη δημιουργία των αντίστοιχων διαγραμμάτων και κατόπιν συγκρίθηκαν με τους υπολογισμούς του προγράμματος REFPROP της NIST [2]. Το πρόγραμμα αυτό είναι σχεδιασμένο ώστε να παρέχει τα πιο ακριβή θερμοφυσικά μεγέθη που είναι διαθέσιμα μέχρι αυτή τη στιγμή για όλα τα ψυκτικά μέσα και τα μίγματα τους. Αυτός είναι και ο λόγος για τον οποίο χρησιμοποιήθηκε ως μέτρο σύγκρισης των αποτελεσμάτων.

Με τον υπολογισμό των σχετικών σφαλμάτων γίνεται φανερό το κατά πόσο το μοντέλο της Peng-Robinson αποκλίνει από την πραγματική συμπεριφορά των μιγμάτων στην αέρια φάση και κρίνεται η καταλληλότητα του.

1.2 Ιστορική αναδρομή

Στη δεκαετία του 1970, οι επιστήμονες άρχισαν να ανησυχούν για ενδεχόμενη βλάβη που μπορούσαν να επιφέρουν ορισμένες χημικές ουσίες στη στιβάδα του όζοντος. Στις αρχές της δεκαετίας του 1980, οι ανησυχίες αυτές επικυρώθηκαν από την ανακάλυψη ότι η ποσότητα του όζοντος στη στρατόσφαιρα πάνω από την Ανταρκτική ήταν ελαττωμένη. Ενώ το όζον δεν εξαφανιζόταν εντελώς σε αυτή την περιοχή, το στρώμα του ήταν τόσο λεπτό που ανάγκασε όλο τον κόσμο πλέον να

μιλάει γι' αυτό. Τα επόμενα χρόνια παρατηρήθηκε επίσης το φαινόμενο του θερμοκηπίου, η δημιουργία δηλαδή ενός ανακλαστικού αέριου στρώματος γύρω από τη γη που επιτρέπει την είσοδο αλλά όχι και την έξοδο της ηλιακής ακτινοβολίας.

Οι συνέπειες που επιφέρουν στον πλανήτη και στον άνθρωπο αυτές οι μεταβολές αποδεικνύονται με το πέρασμα του χρόνου ιδιαίτερα επικίνδυνες. Η καταστροφή της στιβάδας του όζοντος και η συνακόλουθη αύξηση της υπεριώδους ακτινοβολίας (UV) που δέχεται η επιφάνεια της γης, οδηγεί σε δυσμενείς επιπτώσεις για τους ζωικούς και φυτικούς οργανισμούς και σε αλλοιώσεις σε ορισμένα πλαστικά υλικά. Η υπερβολική έκθεση του άνθρωπου στην υπεριώδη ακτινοβολία μπορεί να προκαλέσει πολλά προβλήματα υγείας, συμπεριλαμβανομένης της καταστροφής του δέρματος (καρκίνος του δέρματος, πρόωρη γήρανση), βλάβη στα μάτια (καταρράκτης, κτλ) και καταστολή του ανοσοποιητικού συστήματος. Από την άλλη πλευρά το φαινόμενο του θερμοκηπίου και ο εγκλωβισμός μεγάλης ποσότητας ενέργειας στην ατμόσφαιρα έχει ως επακόλουθο την αύξηση της θερμοκρασίας της γης, την κλιματική αλλαγή και την πρόκληση ακραίων καιρικών φαινομένων.

Λόγω αυτών των κινδύνων κρίθηκε αναγκαία η εξεύρεση μιας εφαρμόσιμης λύσης για τη διαφύλαξη του πλανήτη. Από το 1987, 191 χώρες (σχεδόν όλες οι χώρες του κόσμου) έχουν υπογράψει το Πρωτόκολλο του Μόντρεαλ, μια συνθήκη ορόσημο για την προστασία του περιβάλλοντος. Κύριος στόχος του πρωτοκόλλου είναι η ελάττωση και τελικά η εξάλειψη της παραγωγής και της χρήσης των τεχνητών ουσιών που συμβάλλουν στη μείωση του όζοντος στη στρατόσφαιρα (Ozone Depleting Substances-ODS) . Η αντίστοιχη συνθήκη για τα αέρια του θερμοκηπίου ήρθε αρκετά χρόνια αργότερα με την υπογραφή του Πρωτοκόλλου του Κιότο το 1997. Με αυτό τον τρόπο πιστεύεται ότι η ζημιά μακροπρόθεσμα θα αναστραφεί και θα αποκατασταθεί η ισορροπία του συστήματος.

Η βλάβη που επιφέρει μια ουσία στη στιβάδα του όζοντος καλείται ODP (Ozone Depleting Potential) και ορίζεται ως ο λόγος της απώλειας του όζοντος που επιφέρει μια ποσότητα της αναφερόμενης ουσίας προς την απώλεια που επιφέρει η ίδια ποσότητα του R11, ουσία που προκαλεί τη μεγαλύτερη ζημιά. Ανάλογα περιγράφεται η συνεισφορά των ουσιών στην αύξηση της θερμοκρασίας της γης από τον δείκτη GWP (Global Warming Potential) συγκρινόμενη με την αύξηση της θερμοκρασίας που επιφέρει το διοξείδιο του άνθρακα.

Τα κυριότερα ψυκτικά μέσα που χρησιμοποιήθηκαν μέχρι τα τέλη του '80, όπως τα R11, R12, R22 και διάφοροι άλλοι χλωροφθοράνθρακες (CFCs) ήταν ODSs, οπότε οι επιστήμονες άρχισαν σιγά σιγά να αναζητούν νέες ουσίες πιο καθαρές για το περιβάλλον. Αρχικά προσανατολίστηκαν στους υδροχλωροφθοράνθρακες (HCFCs) που παρουσίασαν μικρό ODP και συνέχεια στους υδροφθοράνθρακες (HFCs), ουσίες με σχεδόν μηδενικό ODP αλλά σε πολλές περιπτώσεις με αρκετά υψηλό GWP της τάξης 10^2-10^4. Πλέον έχει απαγορευθεί η πώληση των HCFCs και των CFCs και η λύση αναζητάται σε μίγματα HFCs προσπαθώντας την επίτευξη των επιθυμητών ιδιοτήτων που ορίζουν οι ψυκτικές και κλιματιστικές εφαρμογές. Μελετώνται επίσης ως ψυκτικά μέσα κάποια φυσικά αέρια όπως αμμωνία, προπάνιο, βουτάνιο, διοξείδιο του άνθρακα και νερό, τα οποία εμφανίζουν μηδενικό ODP και πολύ χαμηλό GWP.

1.3 Επιλογή των μελετώμενων ψυκτικών μιγμάτων

Το R407b είναι ήδη ευρέως διαδεδομένο στην αγορά ως ψυκτικό μέσο και έχει ήδη διερευνηθεί η συμπεριφορά του, όχι μόνο στην αέρια κατάσταση αλλά και στην υγρή και διφασική περιοχή. Η επιλογή του σε αυτήν την εργασία καθορίστηκε από τρεις βασικούς παράγοντες.

Πρώτον είναι και αυτό τριμερές μίγμα, όπως και το ψυκτικό για το οποίο κατασκευάστηκε ο υπολογιστικός κώδικας. Αποτελείται από τις χημικές ενώσεις R32 /R125/R134a σε αναλογίες 10/70/20 κατά μάζα, εκ των οποίων οι δύο συμμετέχουν και στη σύσταση του άλλου μίγματος. Ο δεύτερος λόγος είναι ότι και τα δύο μίγματα έχουν σαν σκοπό την αντικατάσταση του ίδιου ψυκτικού μέσου (R22), άρα λειτουργούν σε παρόμοια περιοχή θερμοκρασιών και πιέσεων, η οποία και ενδιαφέρει να μελετηθεί. Τέλος, είναι ήδη γνωστό από παλαιότερη έρευνα του εργαστηρίου [3] ότι το μοντέλο της Peng-Robinson προσεγγίζει ικανοποιητικά τις πραγματικές θερμοδυναμικές τιμές του. Με αυτό το σκεπτικό η Peng-Robinson θεωρήθηκε ότι θα δίνει ικανοποιητικά αποτελέσματα και για το R152a/R125/R32.

Το τριμερές μίγμα R152a/R125/R32 μελετήθηκε από την ομάδα του Jiangtao Wu [1] και διερευνήθηκε ως προς την ψυκτική του ικανότητα και τις θερμοκρασίες λειτουργίας του. Από τα αποτελέσματα φαίνεται ικανό να αντικαταστήσει το R22 σε πολλές εφαρμογές. Τα τρία συστατικά του έχουν μηδενικό ODP και χαμηλό σε σχέση με άλλα ψυκτικά GWP [4]. Αναλυτικότερα τα GWP_{100} παίρνουν τις τιμές για το R152a: 122, για το R125: 3450 και για το R32: 543. Ο δείκτης αναφέρεται στα χρόνια μετά τα οποία υπολογίζεται το μέγεθος αυτό. Προς σύγκριση αναφέρονται οι αντίστοιχες τιμές για το R22: ODP=0,0055, GWP_{100}=1780 και για το R407b: ODP=0, GWP_{100}=2285. Το γεγονός αυτό καθιστά την προοπτική χρήσης του ως ψυκτικό μέσο ιδιαίτερα ενδιαφέρουσα.

Αυτός είναι και ο λόγος που επιλέχθηκε το συγκεκριμένο μίγμα. Σύμφωνα με τον Jiangtao Wu [1] το αέριο εξετάστηκε σε διάφορες αναλογίες αναζητώντας τη βέλτιστη, με κριτήριο την κατ' όγκο ψυκτική ικανότητα και τον COP. Ως τέτοια βρέθηκε η 48/18/34 κατά μάζα, η οποία και χρησιμοποιείται στην παρούσα διπλωματική εργασία. Το νέο τριμερές μίγμα έχει παραπλήσιες καμπύλες πιέσεων αερίου σε ευρύ θερμοκρασιακό πεδίο με το R22 και ελαφρά καλύτερες επιδόσεις όσον αφορά την ψυκτική ικανότητα και τον COP. Τα θεωρητικά αυτά χαρακτηριστικά εξετάστηκαν σε πειραματική διάταξη μαζί με το R22, αλλά και άλλα αέρια που μελετώνται ως αντικαταστάτες του R22, όπως το R407c και το R410. Τα αποτελέσματα επιβεβαίωσαν ότι το καινούριο μίγμα υπερέχει των άλλων και ότι παρουσιάζει όντως παρόμοια συμπεριφορά με αυτή του R22 (με κριτήρια πάντα τις παρόμοιες θερμοφυσικές ιδιότητες, την ψυκτική ικανότητα και τον COP).

Στην ίδια εργασία, το R152a/R125/R32 δοκιμάστηκε πειραματικά και αν και βρέθηκε ελαφρά αναφλέξιμο, κρίνεται γενικά ασφαλές στη χρήση. Η διαρροή του στον κύκλο λειτουργίας της πειραματικής διάταξης επέδρασε ελάχιστα στην αλλαγή της σύνθεσής του. Συνεπώς δεν υπήρχαν ιδιαίτερες αλλαγές στις θερμοφυσικές του ιδιότητες και στη γενικότερη ψυκτική του συμπεριφορά. Επιπλέον, ένα πολύ

σημαντικό πλεονέκτημα του μίγματος αυτού είναι ότι έχει παραπλήσιες πιέσεις λειτουργίας με το R22, οπότε μπορούν να χρησιμοποιηθούν οι ήδη υπάρχοντες συμπιεστές και εγκαταστάσεις (drop-in replacement).

Το σύνολο αυτών των χαρακτηριστικών καθιστούν το R152a/R125/R32 ελκυστικό υποψήφιο αντικαταστάτη του R22, το οποίο μέχρι τώρα είχε βρει εφαρμογές σε πληθώρα ψυκτικών συστημάτων, αλλά με το πέρασμα του χρόνου απομακρύνεται από την αγορά λόγω της βλάβης που επιφέρει στο περιβάλλον.

Στην υπάρχουσα εργασία υπολογίζονται τα θερμοφυσικά μεγέθη του μίγματος στην αέρια φάση μόνο, πληροφορίες που είναι απαραίτητες για ενδεχόμενη μελλοντική χρήση του ως ψυκτικό, αλλά όχι πλήρως καθώς απαιτείται η εύρεση των αντίστοιχων μεγεθών στην υγρή και στη διφασική περιοχή. Το χρονικό πλαίσιο ενός εξαμήνου της διπλωματικής εργασίας δεν επέτρεψε την περαιτέρω διερεύνηση των ψυκτικών σε αυτές τις περιοχές.

Ονοματολογία

A,B,C,D	συντελεστές ιδανικής θερμοχωρητικότητας υπό σταθερή πίεση
α_c, $\alpha_{1,i}$	συντελεστές Peng-Robinson (kPa cm^6/mol^2)
$\alpha_{2,i}$, $\alpha_{3,i}$	συντελεστές Peng-Robinson
b	συντελεστής Peng-Robinson (cm^3/mol)
c_p	ειδική θερμοχωρητικότητα υπό σταθερή πίεση (kJ/kg K)
c_v	ειδική θερμοχωρητικότητα υπό σταθερό όγκο (kJ/kg K)
h	ειδική ενθαλπία (kJ/kg)
k	συντελεστής αλληλεπίδρασης
$k_{p,T}$, $k_{p,v}$, $k_{T,v}$	ισεντροπικοί εκθέτες
l	συντελεστής αλληλεπίδρασης
M	μοριακό βάρος (kg/kmol)
p	πίεση (kPa)
R	παγκόσμια σταθερά αερίων (kJ/kmol K)
s	ειδική εντροπία (kJ/kg K)
T	θερμοκρασία (K)
V	ειδικός όγκος (cm^3/mol)
v	ειδικός όγκος (m^3/kg)
x	κατά mol σύσταση
w	κατά μάζα σύσταση
Z	συντελεστής συμπιεστότητας
α	ταχύτητα ήχου σε υπέρθερμο ατμό (m/sec)
γ	λόγος c_p / c_v
ρ	πυκνότητα (mol/cm^3)
ω	συντελεστής εκκεντρότητας

Κάτω δείκτες		
b	σημείο βρασμού	
c	κρίσιμο σημείο	
g	μέγεθος εκφρασμένο σε S.I.	
m	μίγματος	
r	ανηγμένο μέγεθος	
0	σημείο αναφοράς	
1	R32 για μίγμα R407b	R152a για μίγμα R152a/R125/R32
2	R125 για μίγμα R407b	R125 για μίγμα R152a/R125/R32
3	R134a για μίγμα R407b	R32 για μίγμα R152a/R125/R32

Άνω δείκτες	
id	ιδανικό αέριο
R	υπολειπόμενα μεγέθη

Κεφάλαιο 2ο :

2.1 Η καταστατική εξίσωση Peng-Robinson και οι υπολογισμοί των συντελεστών της

Στην υπέρθερμη περιοχή για τα πραγματικά αέρια, ο καθορισμός ενός καταστατικού μεγέθους απαιτεί την γνώση των άλλων δύο (p=p(T,v)). Σύμφωνα με τη θερμοδυναμική η καταστατική εξίσωση Peng-Robinson [5] έχει τη μορφή:

$$p = \frac{R_g T}{v - b_g} - \frac{a_g(T)}{v^2 + 2vb_g - b_g^{\,2}} = \frac{1000RT}{V - b} - \frac{a(T)}{V^2 + 2Vb - b^2} \qquad (2.1.1)$$

Το 1000 είναι απλά παράγοντας διόρθωσης μονάδων, ώστε σύμφωνα με τον προαναφερθέντα συμβολισμό και επιλογή μονάδων η πίεση τελικά να εκφράζεται σε kPa. Για τα καθαρά συστατικά του μίγματος ισχύει [3]:

$$a_i(T) = a_{c,i}\left[1 + a_{1i}\left(1 - \sqrt{T_{ri}}\right) + a_{2,i}\left(1 - \sqrt{T_{ri}}\right)^2 + a_{3,i}\left(1 - \sqrt{T_{ri}}\right)^3\right]^2 \qquad (2.1.2)$$

$$T_{ri} = \frac{T}{T_{ci}} \qquad (2.1.3)$$

Όπου T_{ci} και T_{ri} η κρίσιμη και η ανηγμένη θερμοκρασία του κάθε συστατικού. Οι μεταβλητές a και b του μίγματος καθορίζονται από τους ακόλουθους κανόνες ανάμιξης :

$$a_m(T) = \sum_{i=1}^{3}\sum_{j=1}^{3} x_i x_j a_{ij} + 3x_1 x_2 x_3 a_{123} \qquad (2.1.4)$$

$$b_m = \sum_{i=1}^{3}\sum_{j=1}^{3} x_i x_j b_{ij} \qquad (2.1.5)$$

όπου x_i η κατά mol σύσταση του συστατικού στο μίγμα και:

$$a_{ij} = a_{ji} = \sqrt{a_i a_j}\left(1 - k_{ij}\right) \qquad (2.1.6)$$

$$b_{ij} = b_{ji} = \frac{b_i + b_j}{2}\left(1 - l_{ij}\right) \quad i \neq j \qquad (2.1.7)$$

$$a_{123} = k_{123}\sqrt[3]{a_1 a_2 a_3} \tag{2.1.8}$$

όπου οι μεταβλητές k_{ij}, k_{123} και l_{ij} είναι διορθωτικοί παράγοντες και εκφράζουν την αλληλεπίδραση των συστατικών η οποία οδηγεί σε απόκλιση από την Peng-Robinson. Οι τιμές τους προσδιορίζονται πειραματικά για κάθε διμερές μίγμα και μέχρι αυτή τη στιγμή για τα k_{ij} και k_{123} προτείνονται συναρτήσεις γραμμικές ως προς τη θερμοκρασία και για τα l_{ij} απλές τιμές. Οι προσεγγίσεις αυτές είναι γενικά ικανοποιητικές.

Για τον προσδιορισμό των θερμοδυναμικών μεγεθών του υπέρθερμου ατμού είναι αναγκαία η εύρεση των ποσοτήτων $\partial a_m(T)/\partial T$ και $\partial^2 a_m(T)/\partial T^2$. Με χρήση των μαθηματικών προκύπτει:

$$\frac{\partial a_m}{\partial T} = \sum_{i=1}^{3}\sum_{j=1}^{3} x_i x_j \frac{\partial a_{ij}}{\partial T} + 3x_1 x_2 x_3 \frac{\partial a_{123}}{\partial T} \tag{2.1.9}$$

$$\frac{\partial a_i}{\partial T} = -\frac{a_c}{T_c\sqrt{T_r}}\left(1 + a_1\left(1-\sqrt{T_r}\right) + a_2\left(1-\sqrt{T_r}\right)^2 + a_3\left(1-\sqrt{T_r}\right)^3\right) \left(a_1 + 2a_2\left(1-\sqrt{T_r}\right) + 3a_3\left(1-\sqrt{T_r}\right)^2\right) \tag{2.1.10}$$

$$\frac{\partial a_{ij}}{\partial T} = \frac{1-k_{ij}}{2}\left(\frac{\partial a_i}{\partial T}\sqrt{\frac{a_j}{a_i}} + \frac{\partial a_j}{\partial T}\sqrt{\frac{a_i}{a_j}}\right) - \sqrt{a_i a_j}\frac{\partial k_{ij}}{\partial T} \tag{2.1.11}$$

$$\frac{\partial a_{123}}{\partial T} = \frac{k_{123}\left(a_1 a_2 a_3\right)^{-2/3}}{3}\left(a_1 a_2 \frac{\partial a_3}{\partial T} + a_1 a_3 \frac{\partial a_2}{\partial T} + a_2 a_3 \frac{\partial a_1}{\partial T}\right) + \frac{\partial k_{123}}{\partial T}\left(a_1 a_2 a_3\right)^{1/3} \tag{2.1.12}$$

$$\frac{\partial^2 a_m}{\partial T^2} = \sum_{i=1}^{3}\sum_{j=1}^{3} x_i x_j \frac{\partial^2 a_{ij}}{\partial T^2} + 3x_1 x_2 x_3 \frac{\partial^2 a_{123}}{\partial T^2} \tag{2.1.13}$$

$$\begin{aligned}\frac{\partial^2 a_i}{\partial T^2} = \frac{a_c}{2T_r T_c^{\,2}}\Bigg[& \frac{1}{\sqrt{T_r}}\left(1 + a_1\left(1-\sqrt{T_r}\right) + a_2\left(1-\sqrt{T_r}\right)^2 + a_3\left(1-\sqrt{T_r}\right)^3\right) \\ & \cdot \left(a_1 + 2a_2\left(1-\sqrt{T_r}\right) + 3a_3\left(1-\sqrt{T_r}\right)^2\right) \\ & + \left(1 + a_1\left(1-\sqrt{T_r}\right) + a_2\left(1-\sqrt{T_r}\right)^2 + a_3\left(1-\sqrt{T_r}\right)^3\right)\left(2a_2 + 6a_3\left(1-\sqrt{T_r}\right)\right) \\ & + \left(a_1 + 2a_2\left(1-\sqrt{T_r}\right) + 3a_3\left(1-\sqrt{T_r}\right)^2\right)^2\Bigg]\end{aligned} \tag{2.1.14}$$

$$\begin{aligned}\frac{\partial^2 a_{ij}}{\partial T^2} = \frac{1-k_{ij}}{4\sqrt{a_i a_j}}\Bigg(& 2\frac{\partial a_i}{\partial T}\frac{\partial a_j}{\partial T} - \frac{a_i}{a_j}\left(\frac{\partial a_j}{\partial T}\right)^2 - \frac{a_j}{a_i}\left(\frac{\partial a_i}{\partial T}\right)^2 + 2a_i\frac{\partial^2 a_j}{\partial T^2} + 2a_j\frac{\partial^2 a_i}{\partial T^2}\Bigg) \\ & - \frac{\partial k_{ij}}{\partial T}\left(\frac{\partial a_i}{\partial T}\sqrt{\frac{a_j}{a_i}} + \frac{\partial a_j}{\partial T}\sqrt{\frac{a_i}{a_j}}\right) - \cancelto{0}{\frac{\partial^2 k_{ij}}{\partial T^2}}\sqrt{a_i a_j}\end{aligned} \tag{2.1.15}$$

$$\begin{aligned}\frac{\partial^2 a_{123}}{\partial T^2} = & \frac{2\frac{\partial k_{123}}{\partial T}\left(a_1 a_2 a_3\right)^{-2/3}}{3}\left(a_1 a_2\frac{\partial a_3}{\partial T} + a_1 a_3\frac{\partial a_2}{\partial T} + a_2 a_3\frac{\partial a_1}{\partial T}\right) \\ & - \frac{2k_{123}\left(a_1 a_2 a_3\right)^{-5/3}}{9}\left(a_1 a_2\frac{\partial a_3}{\partial T} + a_1 a_3\frac{\partial a_2}{\partial T} + a_2 a_3\frac{\partial a_1}{\partial T}\right)^2 \\ & + \frac{k_{123}\left(a_1 a_2 a_3\right)^{-2/3}}{3}\Bigg(a_1 a_2\frac{\partial^2 a_3}{\partial T^2} + a_1 a_3\frac{\partial^2 a_2}{\partial T^2} + a_2 a_3\frac{\partial^2 a_1}{\partial T^2} \\ & + 2a_1\frac{\partial a_2}{\partial T}\frac{\partial a_3}{\partial T} + 2a_2\frac{\partial a_1}{\partial T}\frac{\partial a_3}{\partial T} + 2a_3\frac{\partial a_1}{\partial T}\frac{\partial a_2}{\partial T}\Bigg) \\ & + \cancelto{0}{\frac{\partial^2 k_{123}}{\partial T^2}}\left(a_1 a_2 a_3\right)^{1/3}\end{aligned} \tag{2.1.16}$$

2.2 *Μαθηματική διατύπωση των φυσικών και θερμοδυναμικών ιδιοτήτων*

Σε αυτό το μέρος θα περιγραφούν όλες οι μαθηματικές εξισώσεις που χρησιμοποιήθηκαν για να υπολογιστούν όλα τα επιθυμητά θερμοφυσικά μεγέθη των δύο μιγμάτων.

Το μοριακό βάρος ακολουθεί τον απλό κανόνα μίξης [6]:

$$M = \sum_{i=1}^{3} x_i M_i \tag{2.2.1}$$

Η γραμμομοριακή σύσταση του κάθε συστατικού βρίσκεται [6]:

$$x_1 = \frac{w_1}{w_1 + w_2\left(M_1/M_2\right) + w_3\left(M_1/M_3\right)} \tag{2.2.2}$$

$$x_2 = \frac{w_2}{w_2 + w_1\left(M_2/M_1\right) + w_3\left(M_2/M_3\right)} \tag{2.2.3}$$

$$x_3 = \frac{w_3}{w_3 + w_1\left(M_3/M_1\right) + w_2\left(M_3/M_2\right)} \tag{2.2.4}$$

Η ειδική σταθερά του αέριου μίγματος ορίζεται [6]:

$$R_g = \frac{R}{M} \tag{2.2.5}$$

Οι κρίσιμες τιμές θερμοκρασίας, πίεσης και όγκου υπολογίστηκαν με δύο διαφορετικούς τρόπους. Ο πρώτος είναι αυτός που προτείνεται ειδικά για την Peng-Robinson [5] και χρησιμοποιεί τις σταθερές μίγματος αυτής και ο δεύτερος αυτός που προτείνεται για την καταστατική εξίσωση Lee-Kesler [5]. Χρησιμοποιήθηκαν δύο τρόποι υπολογισμού διότι κατά τη διάρκεια της εργασίας παρατηρήθηκε ότι η Peng-Robinson, η οποία μελετάται, δίνει κρίσιμα μεγέθη με μεγάλα σφάλματα. Γι' αυτό αποφασίστηκε ο υπολογισμός τους όπως προτείνεται για την Lee-Kesler EoS, η οποία όπως αποδείχθηκε δίνει αποδεκτές τιμές για τα κρίσιμα μεγέθη.

Η κρίσιμη τιμή της θερμοκρασίας με την πρώτη μέθοδο δίνεται από μια επαναληπτική διαδικασία, η οποία λαμβάνει υπ' όψιν τις κρίσιμες τιμές θερμοκρασίας και πίεσης των συστατικών αλλά όχι και των κρίσιμων όγκων.

$$T_{cm} = \frac{a_m\left(T_{cm}^{Old}\right)}{0,457235 \cdot 10^6 R^2 \sum_{i=1}^{3} x_i \frac{T_{ci}}{p_{ci}}} \qquad (2.2.6)$$

Το $a_m\left(T_{cm}\right)$ υπολογίζεται θέτοντας $T = T_{cm}$ στη σχέση υπολογισμού του a_i και ακολουθώντας τους κανόνες μίξης για την Peng-Robinson που περιγράφηκαν στο πρώτο μέρος του κεφαλαίου αυτού:

$$a_i\left(T_{cm}\right) = a_{c,i}\left[1 + a_{1i}\left(1 - \sqrt{T_{ri,cm}}\right) + a_{2,i}\left(1 - \sqrt{T_{ri,cm}}\right)^2 + a_{3,i}\left(1 - \sqrt{T_{ri,cm}}\right)^3\right]^2 \qquad (2.2.7)$$

$$T_{ri,cm} = \frac{T_{cm}}{T_{ci}} \qquad (2.2.8)$$

Ο υπολογισμός της κρίσιμης πίεσης του αερίου έγινε από την απλή σχέση μίξης:

$$p_{cm} = \frac{T_{cm}}{\sum_{i=1}^{3} x_i \frac{T_{ci}}{p_{ci}}} \qquad (2.2.9)$$

Για τον κρίσιμο όγκο δεν χρησιμοποιήθηκε κάποιος κανόνας μίξης που να συνυπολογίζει τους κρίσιμους όγκους των συστατικών, αλλά η εξίσωση Peng-Robinson (2.1.1). Γι' αυτό χρησιμοποιήθηκε επαναληπτική διαδικασία στον κώδικα, λόγω της τριτοβάθμιας εξίσωσης που προκύπτει για τον κρίσιμο όγκο. Έπειτα από δοκιμές διαφόρων επαναληπτικών σχέσεων καταλήξαμε στη βέλτιστη:

$$V_{cm} = \frac{1000RT_{cm}}{p_{cm}} - \frac{a_m\left(T_{cm}\right)}{\left(b_m + V_{cm}^{0ld}\frac{V_{cm}^{0ld} + b_m}{V_{cm}^{0ld} - b_m}\right)p_{cm}} + b_m \qquad (2.2.10)$$

Ο δεύτερος τρόπος υπολογισμού των κρίσιμων μεγεθών περιλαμβάνει τις ακόλουθες εκφράσεις [5]:

$$T_{cm} = \frac{1}{V_{cm}^{1/4}} \sum_{i=1}^{3}\sum_{j=1}^{3} x_i x_j V_{cij}^{1/4} T_{cij} \qquad (2.2.11)$$

$$V_{cm} = \sum_{i=1}^{3}\sum_{j=1}^{3} x_i x_j V_{cij} \qquad (2.2.12)$$

$$T_{cij} = \left(T_{ci} T_{cj}\right)^{1/2} \tag{2.2.13}$$

$$V_{cij} = \frac{1}{8}\left(V_{ci}^{1/3} + V_{cj}^{1/3}\right)^3 \tag{2.2.14}$$

$$p_{cm} = \frac{\left(0.2905 - 0.085\omega_m\right) RT_{cm}}{V_{cm}} \tag{2.2.15}$$

όπου ω ο συντελεστής εκκεντρότητας, όρος που αναφέρεται στη μη σφαιρικότητα του μορίου της ουσίας και ο οποίος δίνεται από την έκφραση [7]:

$$\omega = \frac{-\ln p_c - 5.92714 + 6.09648e^{-1} + 1.28862\ln e - 0.169347e^6}{15.2518 - 15.6875e^{-1} - 13.472\ln e + 0.43577e^6} \tag{2.2.16}$$

$$e = \frac{T_b}{T_c} \tag{2.2.17}$$

Η ειδική θερμοχωρητικότητα υπό σταθερή πίεση σε χαμηλές πιέσεις (ιδανικό αέριο) είναι συνάρτηση της θερμοκρασίας . Η πιο συνηθισμένη μορφή της είναι η πολυωνυμική:

$$c_p^{id} = A + BT + CT^2 + DT^3 \tag{2.2.18}$$

οι σταθερές της οποίας εξαρτώνται από τη χημική σύσταση του συστατικού. Για το R407b χρησιμοποιείται η προτεινόμενη έκφραση [8]:

$$c_p^{id} = A + BT + CT^2 + \frac{D}{T} \tag{2.2.19}$$

Σε περίπτωση μιγμάτων για τη σταθερά Α ακολουθείται ο απλός κανόνας μίξης:

$$A = \sum_{i=1}^{3} x_i A_i \tag{2.2.20}$$

όμοια βρίσκονται οι σταθερές B,C και D του μίγματος.

Για κάθε πραγματικό αέριο μπορούμε να θεωρήσουμε ότι η τιμή της επιθυμητής μεταβλητής αποτελείται από το άθροισμα της ιδανικής τιμής τελείου αερίου και της υπολειπόμενης τιμής αυτού [6]. Έτσι για την εντροπία θεωρούμε:

$$s = s^{id} + s^R \tag{2.2.21}$$

Ως γνωστόν τα πραγματικά αέρια συμπεριφέρονται ως τέλεια σε πολύ μικρές πιέσεις ή ισοδύναμα σε πολύ μικρές πυκνότητες, όπου οι αλληλεπιδράσεις των μορίων είναι αμελητέες. Οπότε οι ιδανικές τιμές ορίζονται από τη θερμοκρασία και τις συνθήκες αναφοράς, ενώ τα υπολειπόμενα μεγέθη εξαρτώνται από την πίεση ή τον όγκο κατ' επιλογήν. Το σημείο αναφοράς επιλέγεται αυθαίρετα αφού η εντροπία δεν είναι καταστατικό μέγεθος και έχει νόημα μόνο η διαφορά της μεταξύ δύο καταστάσεων . Σύμφωνα με αυτά:

$$s^{id} = s_0 + \int_{T_o}^{T} \left(\frac{\partial s^{id}}{\partial T} \right)_v dT \qquad (2.2.22)$$

$$s^R = \int_0^{\rho_g} \left(\frac{\partial s^R}{\partial \rho_g} \right)_T d\rho_g = \int_0^{\rho_g} \left(\left(\frac{\partial s}{\partial \rho_g} \right)_T - \left(\frac{\partial s^{id}}{\partial \rho_g} \right)_T \right) d\rho_g \qquad (2.2.23)$$

Από τη θερμοδυναμική [1] ισχύουν:

$$\left(\frac{\partial s}{\partial v} \right)_T = \left(\frac{\partial p}{\partial T} \right)_v \Rightarrow -\rho_g^2 \left(\frac{\partial s}{\partial \rho_g} \right)_T = \left(\frac{\partial p}{\partial T} \right)_{\rho_g} \quad \text{(σχέση Maxwell)} \qquad (2.2.24)$$

$$\left(\frac{\partial s}{\partial p} \right)_T = -\left(\frac{\partial v}{\partial T} \right)_p \quad \text{(σχέση Maxwell)} \qquad (2.2.25)$$

$$du = Tds - pdv \Rightarrow \left(\frac{\partial u}{\partial T} \right)_v = T \left(\frac{\partial s}{\partial T} \right)_v - p \cancelto{0}{\left(\frac{\partial v}{\partial T} \right)_v} \Rightarrow \left(\frac{\partial s}{\partial T} \right)_v = \frac{c_v}{T} \qquad (2.2.26)$$

και αφού θεωρηθεί ότι το ρ_g είναι εκφρασμένο σε m^3/kg ενώ το ρ σε cm^3/mol

$$\rho_g = 1000 M \rho \qquad (2.2.27)$$

από τις σχέσεις 2.2.18, 2.2.22, 2.2.26 και 2.2.27 συνεπάγεται:

$$s^{id} = s_0 + \int_{T_o}^{T} \left(\frac{c_v}{T} \right)^{id} dT = s_0 + \int_{T_o}^{T} \left(\frac{c_p}{T} \right)^{id} dT - R_g \ln \left| \frac{p}{p_o} \right| \Rightarrow$$

$$s^{id} = s_0 + \left(A \ln T + BT + \frac{CT^2}{2} + \frac{DT^3}{3} \right) - \left(A \ln T_o + BT_o + \frac{CT_o^2}{2} + \frac{DT_o^3}{3} \right) - R_g \ln \left| \frac{\rho RT}{p_o} \right| \qquad (2.2.28)$$

(χρησιμοποιούνται οι εκφράσεις c_p^{id} του R152a/R125/R32 (2.2.18), παρόμοια ισότητα αναλογεί στο R407b χρησιμοποιώντας τη 2.2.19). Από 2.2.23, 2.2.24 και 2.2.27:

$$s^R = \int_0^{\rho_g} \frac{1}{\rho_g^2}\left(\left(\frac{\partial p^{id}}{\partial T}\right)_{\rho_g} - \left(\frac{\partial p}{\partial T}\right)_{\rho_g}\right) d\rho_g = \frac{1}{1000M}\int_0^{\rho} \frac{1}{\rho^2}\left(\frac{p}{T} - \left(\frac{\partial p}{\partial T}\right)_{\rho}\right) d\rho \Rightarrow$$

$$s^R = \frac{1}{1000M}\int_0^{\rho}\left[\frac{R}{\rho} - \frac{1}{\rho^2}\left(\frac{\partial p}{\partial T}\right)_{\rho}\right] d\rho \qquad (2.2.29)$$

Παραγωγίζοντας ως προς τη θερμοκρασία την εξίσωση Peng-Robinson (2.1.1):

$$\left(\frac{\partial p}{\partial T}\right)_{\rho} = \frac{1000R}{V - b_m} - \frac{\dfrac{\partial a_m(T)}{\partial T}}{V^2 + 2Vb_m - b_m^2} \overset{(\rho = 1/v)}{\Rightarrow} \qquad (2.2.30)$$

$$\frac{R}{\rho} - \frac{1}{\rho^2}\left(\frac{\partial p}{\partial T}\right)_{\rho} = \frac{R}{\rho} - \frac{1000RT}{\rho - b_m\rho^2} + \frac{\dfrac{\partial a_m(T)}{\partial T}}{1 + 2b_m\rho - b_m^2\rho^2} \Rightarrow$$

$$\int_0^{\rho}\left[\frac{R}{\rho} - \frac{T}{\rho^2}\left(\frac{\partial p}{\partial T}\right)_{\rho}\right] d\rho = \int_0^{\rho}\left[\frac{R}{\rho} - \frac{1000RT}{\rho - b_m\rho^2} + \frac{\dfrac{\partial a_m(T)}{\partial T}}{1 + 2b_m\rho - b_m^2\rho^2}\right] d\rho \qquad (2.2.31)$$

ισχύει η ιδιότητα:

$$\int_0^x \frac{1}{a\,x^2 + b\,x + c}\,dx = \left[\frac{1}{\left(b^2 - 4ac\right)^{1/2}} \ln \frac{2ax + b - \left(b^2 - 4ac\right)^{1/2}}{2ax + b + \left(b^2 - 4ac\right)^{1/2}}\right]_0^x, b^2 \succ 4\,a\,c, a \neq 0 \qquad (2.2.32)$$

όπου:

$a = 1,\ b = 2\,b_m\ ,\ c = -b_m^2$

Άρα:

$$\int_0^{\rho} \frac{1}{1 + 2b_m\rho - b_m^2\rho^2} d\rho = \frac{1}{\sqrt{8}b_m} \ln \frac{-2b_m^2\rho + 2b_m - \sqrt{8}b_m}{-2b_m^2\rho + 2b_m + \sqrt{8}b_m}\Bigg|_0^{\rho} \qquad (2.2.33)$$

οπότε η υπολειπόμενη εντροπία προσδιορίζεται από 2.2.29, 2.2.31 και 2.2.33:

$$s^R = \frac{1}{1000M}\left[R\ln\left|1-b_m\rho\right| + \frac{\partial a_m(T)}{\partial T}\frac{1}{\sqrt{8}b_m}\left(\ln\left|\frac{-b_m\rho+1-\sqrt{2}}{-b_m\rho+1+\sqrt{2}}\right| - \ln\left|\frac{1-\sqrt{2}}{1+\sqrt{2}}\right|\right)\right] \quad (2.2.34)$$

Από 2.2.28 και 2.2.34 η συνολική εντροπία διαμορφώνεται τελικά:

$$\begin{aligned} s = s_0 &+ \left(A\ln T + BT + \frac{CT^2}{2} + \frac{DT^3}{3}\right) - \left(A\ln T_o + BT_o + \frac{CT_o^2}{2} + \frac{DT_o^3}{3}\right) - R_g \ln\left|\frac{\rho RT}{p_o}\right| \\ &+ \frac{1}{1000M}\left[R\ln\left|1-b_m\rho\right| + \frac{\partial a_m(T)}{\partial T}\frac{1}{\sqrt{8}b_m}\left(\ln\left|\frac{-b_m\rho+1-\sqrt{2}}{-b_m\rho+1+\sqrt{2}}\right| - \ln\left|\frac{1-\sqrt{2}}{1+\sqrt{2}}\right|\right)\right] \end{aligned} \quad (2.2.35)$$

Αντίστοιχα για την εσωτερική ενέργεια ισχύουν :

$$u = u^{id} + u^R \quad (2.2.36)$$

$$u^{id} = u_0 + \int_{T_o}^{T}\left(\frac{\partial u^{id}}{\partial T}\right)_v dT = u_0 + \int_{T_o}^{T} c_v dT = u_0 + \int_{T_o}^{T} c_p dT - R_g\left(T-T_o\right) \overset{(2.2.18)}{\Rightarrow}$$

$$u^{id} = u_0 + \left(AT + \frac{BT^2}{2} + \frac{CT^3}{3} + \frac{DT^4}{4}\right) - \left(AT_o + \frac{BT_o^2}{2} + \frac{CT_o^3}{3} + \frac{DT_o^4}{4}\right) - R_g\left(T-T_o\right) \quad (2.2.37)$$

$$u^R = \int_0^{\rho_g}\left(\frac{\partial u^R}{\partial \rho_g}\right)_T d\rho_g = \int_0^{\rho_g}\left(\left(\frac{\partial u}{\partial \rho_g}\right)_T - \left(\frac{\partial u^{id}}{\partial \rho_g}\right)_T\right) d\rho_g \quad (2.2.38)$$

Από τη θερμοδυναμική ανάλυση βρίσκεται ότι:

$$du = Tds - pdv = Tds + \frac{p}{\rho_g^{\,2}}d\rho_g \Rightarrow \left(\frac{\partial u}{\partial \rho_g}\right)_T = T\left(\frac{\partial s}{\partial \rho_g}\right)_T + \frac{p}{\rho_g^2}\cancel{\left(\frac{\partial \rho_g}{\partial \rho_g}\right)_T}^{\,1} \quad (2.2.39)$$

Και σύμφωνα με τις ισότητες 2.2.24, 2.2.27, 2.2.38 και 2.2.39:

$$u^R = \int_0^{\rho_g} \left(T\left(\frac{\partial s^R}{\partial \rho_g}\right)_T + \left(\frac{p}{\rho_g^2}\right)^R \right) d\rho_g = \int_0^{\rho_g} \left(T\left(\frac{\partial s}{\partial \rho_g}\right)_T + \left(\frac{p}{\rho_g^2}\right) - \left(T\left(\frac{\partial s^{id}}{\partial \rho_g}\right)_T + \left(\frac{p}{\rho_g^2}\right)^{id} \right) \right) d\rho_g \Rightarrow$$

$$u^R = \int_0^{\rho_g} \left(-\frac{T}{\rho_g^2}\left(\frac{\partial p}{\partial T}\right)_{\rho_g} + \left(\frac{p}{\rho_g^2}\right) - \left(\left(-\frac{T}{\rho_g^2}\left(\frac{\partial p}{\partial T}\right)_{\rho_g} \right)^{id} + \left(\frac{p}{\rho_g^2}\right)^{id} \right) \right) d\rho_g \Rightarrow$$

$$u^R = \int_0^{\rho_g} \left(-\frac{T}{\rho_g^2}\left(\frac{\partial p}{\partial T}\right)_{\rho_g} + \left(\frac{p}{\rho_g^2}\right) - \left(\left(-\frac{\rho_g RT}{\rho_g^2} \right)^{id} + \left(\frac{p}{\rho_g^2}\right)^{id} \right)^{0} \right) d\rho_g$$

$$u^R = \int_0^{\rho_g} \left[\frac{p}{\rho_g^2} - \frac{T}{\rho_g^2}\left(\frac{\partial p}{\partial T}\right)_{\rho_g} \right] d\rho_g \Rightarrow$$

$$u^R = \frac{1}{1000M} \int_0^{\rho} \left[\frac{p}{\rho^2} - \frac{T}{\rho^2}\left(\frac{\partial p}{\partial T}\right)_{\rho} \right] d\rho \qquad (2.2.40)$$

Από τη σχέση 2.2.30 σχηματίζεται:

$$\frac{p}{\rho^2} - \frac{T}{\rho^2}\left(\frac{\partial p}{\partial T}\right)_{\rho} = \frac{1000RT}{\rho - b_m\rho^2} - \frac{a_m(T)}{1 + 2b_m\rho - b_m^2\rho^2} - \frac{1000RT}{\rho - b_m\rho^2} + \frac{T\dfrac{\partial a_m(T)}{\partial T}}{1 + 2b_m\rho - b_m^2\rho^2} \Rightarrow \qquad (2.2.41)$$

$$\int_0^{\rho} \left[\frac{p}{\rho^2} - \frac{T}{\rho^2}\left(\frac{\partial p}{\partial T}\right)_{\rho} \right] d\rho = \int_0^{\rho} \left[\frac{T\dfrac{\partial a_m(T)}{\partial T} - a_m(T)}{1 + 2b_m\rho - b_m^2\rho^2} \right] d\rho \qquad (2.2.42)$$

Άρα η υπολειπόμενη εσωτερική ενέργεια δίνεται τελικά από 2.2.33, 2.2.40 και 2.2.42:

$$u^R = \frac{1}{1000M} \frac{T\dfrac{\partial a_m(T)}{\partial T} - a_m(T)}{\sqrt{8}b_m} \left(\ln\left| \frac{-b_m\rho + 1 - \sqrt{2}}{-b_m\rho + 1 + \sqrt{2}} \right| - \ln\left| \frac{1 - \sqrt{2}}{1 + \sqrt{2}} \right| \right) \qquad (2.2.43)$$

Και η εσωτερική ενέργεια διαμορφώνεται από 2.2.36, 2.2.37 και 2.2.43:

$$u = u_0 + \left(AT + \frac{BT^2}{2} + \frac{CT^3}{3} + \frac{DT^4}{4} \right) - \left(AT_o + \frac{BT_o^2}{2} + \frac{CT_o^3}{3} + \frac{DT_o^4}{4} \right) - R_g \left(T - T_o \right) + \frac{1}{1000M} \frac{T \frac{\partial a_m(T)}{\partial T} - a_m(T)}{\sqrt{8} b_m} \left(\ln \left| \frac{-b_m \rho + 1 - \sqrt{2}}{-b_m \rho + 1 + \sqrt{2}} \right| - \ln \left| \frac{1 - \sqrt{2}}{1 + \sqrt{2}} \right| \right) \quad (2.2.44)$$

Η ενθαλπία ορίζεται από τη ακόλουθη σχέση [9]

$$h = u + \frac{p}{\rho_g} \Rightarrow \quad (2.2.45)$$

και με τη βοήθεια των 2.2.27, 2.2.44, 2.2.45 εκφράζεται:

$$h = u_o + \left(AT + \frac{BT^2}{2} + \frac{CT^3}{3} + \frac{DT^4}{4} \right) - \left(AT_o + \frac{BT_o^2}{2} + \frac{CT_o^3}{3} + \frac{DT_o^4}{4} \right) - R_g \left(T - T_o \right) + \frac{1}{1000M} \frac{p}{\rho} + \frac{1}{1000M} \frac{T \frac{\partial a_m(T)}{\partial T} - a_m(T)}{\sqrt{8} b_m} \left(\ln \left| \frac{-b_m \rho + 1 - \sqrt{2}}{-b_m \rho + 1 + \sqrt{2}} \right| - \ln \left| \frac{1 - \sqrt{2}}{1 + \sqrt{2}} \right| \right) \quad (2.2.46)$$

Η ειδική θερμοχωρητικότητα υπό σταθερό όγκο προκύπτει από παραγώγιση της 2.2.44:

$$c_v = \left(\frac{\partial u}{\partial T} \right)_v \Rightarrow$$

$$c_v = A + BT + CT^2 + DT^3 - R_g + \frac{1}{1000M} \frac{T \frac{\partial^2 a_m(T)}{\partial T^2}}{\sqrt{8} b_m} \left(\ln \left| \frac{-b_m \rho + 1 - \sqrt{2}}{-b_m \rho + 1 + \sqrt{2}} \right| - \ln \left| \frac{1 - \sqrt{2}}{1 + \sqrt{2}} \right| \right) \quad (2.2.47)$$

Για τον προσδιορισμό της ειδικής θερμοχωρητικότητας υπό σταθερή πίεση η θερμοδυναμική δίνει:

$$ds = \left(\frac{\partial s}{\partial T} \right)_v dT + \left(\frac{\partial s}{\partial v} \right)_T dv \quad \Rightarrow \quad ds = \frac{c_v}{T} dT + \left(\frac{\partial s}{\partial v} \right)_T dv \quad \Rightarrow \quad Tds = c_v dT + T \left(\frac{\partial s}{\partial v} \right)_T dv \quad \Rightarrow$$

$$T \left(\frac{\partial s}{\partial T} \right)_p = c_v + T \left(\frac{\partial s}{\partial v} \right)_T \left(\frac{\partial v}{\partial T} \right)_p \overset{(2.2.25)}{\Rightarrow} \quad c_p = c_v - T \left(\frac{\partial s}{\partial v} \right)_T \left(\frac{\partial s}{\partial p} \right)_T \quad \Rightarrow \quad c_p = c_v - T \left(\frac{\partial s}{\partial v} \right)_T^2 \left(\frac{\partial v}{\partial p} \right)_T \quad \Rightarrow$$

$$c_p = c_v - T\left(\frac{\partial p}{\partial T}\right)_v^2 \left(\frac{\partial p}{\partial v}\right)_T^{-1} \qquad \Rightarrow$$

$$c_p = c_v - \frac{T}{1000M}\left(\frac{\partial p}{\partial T}\right)_V^2 \left(\frac{\partial p}{\partial V}\right)_T^{-1} \tag{2.2.48}$$

Με παραγώγιση της Peng Robinson (2.1.1) ως προς τον ειδικό όγκο προκύπτει:

$$\left(\frac{\partial p}{\partial V}\right)_T = -\frac{1000RT}{\left(V-b_m\right)^2} + \frac{a_m(T)\left(2V-2b_m\right)}{\left(V^2+2Vb_m-b_m^2\right)^2} \tag{2.2.49}$$

Άρα η τελική έκφραση για την ειδική θερμοχωρητικότητα υπό σταθερή πίεση προκύπτει από 2.2.18, 2.2.30, 2.2.48 και 2.2.49:

$$\begin{aligned} c_p = {} & A+BT+CT^2+DT^3-R_g + \frac{T}{1000M}\frac{\frac{\partial^2 a_m(T)}{\partial T^2}}{\sqrt{8}b_m}\left(\ln\left|\frac{-b_m\rho+1-\sqrt{2}}{-b_m\rho+1+\sqrt{2}}\right| - \ln\left|\frac{1-\sqrt{2}}{1+\sqrt{2}}\right|\right) \\ & -\frac{T}{1000M}\left(\frac{1000R}{V-b_m} - \frac{\frac{\partial a_m(T)}{\partial T}}{V^2+2Vb_m-b_m^2}\right)^2 \left(-\frac{1000RT}{\left(V-b_m\right)^2} + \frac{a_m(T)\left(2V-2b_m\right)}{\left(V^2+2Vb_m-b_m^2\right)^2}\right)^{-1} \end{aligned} \tag{2.2.50}$$

Για τους ισεντροπικούς εκθέτες, οι οποίοι ορίζονται από τις σχέσεις:

$$\begin{aligned} & pv^{k_{p,v}} = const. \\ & Tv^{k_{T,v}-1} = const. \\ & p^{1-k_{p,T}}T^{k_{p,T}} = const. \end{aligned} \tag{2.2.51}$$

ισχύουν οι παρακάτω σχέσεις [3] οπότε με τις εκφράσεις 2.1.1, 2.2.47, 2.2.48 και 2.2.49 για τα p, c_v, c_p και $\left(\frac{\partial p}{\partial v}\right)_T$ αντίστοιχα είναι εύκολος ο προσδιορισμός τους:

$$k_{p,v} = -\frac{v}{p}\frac{c_p}{c_v}\left(\frac{\partial p}{\partial v}\right)_T = -\frac{V}{p}\frac{c_p}{c_v}\left(\frac{\partial p}{\partial V}\right)_T \tag{2.2.52}$$

$$k_{T,v} = 1+\frac{v}{c_v}\left(\frac{\partial p}{\partial T}\right)_v = 1+\frac{1}{1000M}\frac{V}{c_v}\left(\frac{\partial p}{\partial T}\right)_V \tag{2.2.53}$$

$$k_{p,T} = \frac{T\left(\frac{\partial p}{\partial T}\right)_v}{T\left(\frac{\partial p}{\partial T}\right)_v + p\left(\frac{c_v}{c_p} - 1\right)} = \frac{T\left(\frac{\partial p}{\partial T}\right)_V}{T\left(\frac{\partial p}{\partial T}\right)_V + p\left(\frac{c_v}{c_p} - 1\right)} \quad (2.2.54)$$

Ο λόγος c_p / c_v υπολογίζεται:

$$\gamma = \frac{c_p}{c_v} \quad (2.2.55)$$

Ο συντελεστής συμπιεστότητας δίνεται από τη σχέση [9]:

$$Z = \frac{R_g T}{pv} = \frac{1000RT}{pV} \quad (2.2.56)$$

Η ταχύτητα του ήχου βρίσκεται [3]:

$$a^2 = -v^2 \frac{c_p}{c_v}\left(\frac{\partial p}{\partial v}\right)_T = -\frac{V^2}{M}\frac{c_p}{c_v}\left(\frac{\partial p}{\partial V}\right)_T \quad (2.2.57)$$

Για την εκπόνηση των διαγραμμάτων των μεγεθών s, h, $k_{p,T}$, $k_{p,v}$,$k_{T,v}$, γ και α για διακεκριμένες τιμές πιέσεων απαιτήθηκε η επίλυση της τριτοβάθμιας ως προς τον όγκο εξίσωσης Peng-Robinson για τους διάφορους συνδυασμούς T,p. Για την επίλυση της δοκιμάστηκαν διάφορες επαναληπτικές μέθοδοι και τελικά χρησιμοποιήθηκε η σχέση:

$$V = \frac{1000RT}{p + \frac{a_m(T)}{V^2 + 2Vb_m - b_m^2}} + b_m \quad (2.2.58)$$

2.3 Περιγραφή και δομή του προγράμματος

Για το κάθε αέριο μίγμα το πρόγραμμα που φτιάχτηκε υπολογίζει τις μεταβλητές p, s, h, c_v, c_p, $k_{p,T}$, $k_{p,v}$, $k_{T,v}$, γ, Z και α σε εύρος θερμοκρασίας 240,15-481,05K και όγκου 0,006-0,6 m^3/kg. Το βήμα θερμοκρασίας επιλέχθηκε 0,1K ενώ το βήμα όγκου ορίστηκε 0,0005 m^3/kg επιτρέποντας τον προσδιορισμό των ιδιοτήτων για 2410 διαφορετικές θερμοκρασίες και 1189 διαφορετικούς όγκους. Επιπλέον το πρόγραμμα υπολογίζει τις κρίσιμες τιμές θερμοκρασίας, πίεσης και όγκου με δύο τρόπους, σύμφωνα με τις προτεινόμενες σχέσεις για τη καταστατική εξίσωση Lee-Kesler και σύμφωνα με τις αντίστοιχες για Peng-Robinson.

Παράλληλα έχει προβλεφθεί ο μηδενισμός των τιμών των μεγεθών για θερμοκρασίες και όγκους που αντιστοιχούν σε διφασική κατάσταση του μίγματος. Ο μηδενισμός γίνεται με σύγκριση του εκάστοτε όγκου με αυτόν του κορεσμένου αερίου για την κάθε θερμοκρασία. Οι τιμές κορεσμού πίεσης και όγκου αντλήθηκαν από το πρόγραμμα REFPROP της NIST [2] και αφού μοντελοποιήθηκαν στο Excell από πολυωνυμικές εξισώσεις $6^{ου}$ βαθμού, στη συνέχεια ενσωματώθηκαν στο κώδικα. Το σφάλμα μοντελοποίησης δεν ξεπέρασε το 0,04% για τις πιέσεις κορεσμού και το 0,02% για τους όγκους κορεσμού. Αυτό το επίπεδο σφάλματος εξασφαλίζει τον μηδενισμό των τιμών της διφασικής περιοχής και μόνων αυτών.

Το πρόγραμμα εκτός από τον υπολογισμό των θερμοφυσικών ιδιοτήτων επιτρέπει την αποθήκευση τους απευθείας στο Excell για 480 συνδυασμούς T,v της υπέρθερμης περιοχής, ενώ με μικρές προσθήκες είναι δυνατή η εξαγωγή οποιουδήποτε επιθυμητού πίνακα σε μορφή αρχείου Excell. Επίσης έχει προβλεφθεί η εκτύπωση στην οθόνη όλων των μεγεθών για τον επιθυμητό συνδυασμό T,v που εισάγεται από τον απλό χρήστη, ο οποίος δεν απαιτείται να έχει προγραμματιστικές γνώσεις. Η εισαχθείσα θερμοκρασία πρέπει να δίνεται σε ^oC και η εισαχθείσα πίεση σε kPa.

Για το κάθε αέριο μίγμα χρησιμοποιήθηκε ένα διαφορετικό κυρίως πρόγραμμα για τον υπολογισμό των μεγεθών και ένα δευτερεύων για την εκπόνηση των επιθυμητών διαγραμμάτων όπου απαιτούνταν οι υπολογισμοί με βάση την επιθυμητή τιμή πίεσης αντί όγκου. Για να επιτευχθεί αυτό χρησιμοποιήθηκε επαναληπτική διαδικασία στο κώδικα για την επίλυση της τριτοβάθμιας εξίσωσης που δημιουργεί η Peng-Robinson. Η επαναληπτική διαδικασία (2.2.57) συνέκλινε στη σωστή τιμή όγκου (η τριτοβάθμια εξίσωση έχει τρεις ακριβώς λύσεις) για όλες τις τιμές της υπέρθερμης περιοχής με αρχικές τιμές V=1950 και με συνθήκη τερματισμού $\left|V^{Old}-V\right|<10^{-6}$, η οποία εξασφαλίζει σφάλματα της τάξης του 0,02%. Οι ίδιες αρχικές τιμές και συνθήκες τερματισμού χρησιμοποιήθηκαν και για την εύρεση του κρίσιμου όγκου (2.2.10) ενώ για τη κρίσιμη θερμοκρασία (2.2.6) χρησιμοποιήθηκε αρχική τιμή T=1 και ως συνθήκη τερματισμού η $\left|T_{cm}^{Old}-T_{cm}\right|<10^{-4}$.

Ο κώδικας που χρησιμοποιήθηκε για την υλοποίηση του προγράμματος παρατίθεται στο Παράρτημα Γ της εργασίας.

Κεφάλαιο 3° :

3.1 Συντελεστές και σταθερές του αέριου μίγματος R407b

Για το R407b οι σταθερές της Peng-Robinson παίρνουν τις τιμές [3]:

Πίνακας 3.1.1: *Οι σταθερές της Peng-Robinson του R407b*

Συστατικό	$\alpha_c \times 10^9$ (kPacm^6/mol^2)	b (cm^3/mol)	α_{1i}	α_{2i}	α_{3i}
R32	0.67426524	39.284975	0.826430	-0.458580	1.012514
R125	1.00183635	60.399575	0.824698	-0.223677	0.907843
R134a	1.09197810	59.732891	0.863418	-0.161165	0.382295

Οι κρίσιμες τιμές θερμοκρασίας, πίεσης και ειδικού όγκου των συστατικών δίνονται [3],[8] :

Πίνακας 3.1.2: *Κρίσιμες και άλλες σταθερές του R407b*

Συστατικό	Κρίσιμη T_c (K)	Κρίσιμη p_c (kPa)	Κρίσιμος V_c (cm^3/mol)	Μ. βάρος (kg/kmol)	Συντελεστής εκκεντρότητας	Σύσταση (κατά μάζα)
R152a	351,55	5830,0	123,8096	52,0235	0,267817	0,10
R125	339,45	3630,6	211,40	120,0215	0,294334	0,70
R32	374,16	4067,0	198,9585	102,0311	0,316055	0,20

Οι τιμές του παρακάτω πίνακα s_o και u_o επιλέγονται έτσι ώστε να ταυτίζονται οι υπολογιζόμενες από τη java τιμές της εντροπίας και της ενθαλπίας με αυτές που δίνει το πρόγραμμα REFPROP για το σημείο p_o=251,325kPa, T_o=273,15K. Οι προτεινόμενες τιμές για τους συντελεστές A, B, C, D του ιδανικού c_p, ο οποίος δίνεται από τη σχέση 2.2.19 φαίνονται στον ίδιο πίνακα [2],[8].

Πίνακας 3.1.3: *Σταθερές αναφοράς και ιδανικού c_p του R407b*

	R407b
p_o (kPa)	251,325
T_o (K)	273,15
s_o (kJ/kg K)	1,700
u_o (kJ/kg)	352,64
A	$4{,}389 \times 10^{-2}$
B	$2{,}892 \times 10^{-3}$
C	$-1{,}766 \times 10^{-6}$
D	18,608

Οι αδιάστατες μεταβλητές k_{ij}, k_{123} και l_{ij}, όπου το i=1 αναφέρεται στο R32, το i=2 στο R125 και το i=3 στο R134a, παίρνουν τις τιμές [3]:

$$
\begin{aligned}
k_{12} &= 0.0025 - 0.00003(T - 273.15) \\
k_{13} &= 0.0030 - 0.00007(T - 273.15) \\
k_{23} &= 0.0028 \\
k_{123} &= -0.0150 - 0.00010(T - 273.15)
\end{aligned}
\qquad (3.1.1)
$$

$$
\begin{aligned}
l_{12} &= 0.02019 \\
l_{13} &= 0.02372 \\
l_{23} &= -0.00777
\end{aligned}
\qquad (3.1.2)
$$

Οι πολυωνυμικές εξισώσεις που μοντελοποιήθηκαν στο Excell για τις καμπύλες κορεσμένου αερίου p_{sat} και v_{sat} με βάση τα στοιχεία του προγράμματος REFPROP φαίνονται παρακάτω. Για τον όγκο χρησιμοποιήθηκαν δύο εξισώσεις χωρίζοντας σε δύο ομάδες τις εισαγόμενες θερμοκρασίες για μεγαλύτερη ακρίβεια. Αυτό έγινε επειδή η σύγκριση για το μηδενισμό των μεγεθών όταν το αέριο βρίσκεται στη διφασική περιοχή γίνεται με βάση τον όγκο κορεσμού. Το ανώτατο σφάλμα μοντελοποίησης είναι 0.005% για το v_{sat} και 0.009% για τη p_{sat}.

$$
\begin{aligned}
p_{sat} = {} & 1.49896432939667 \cdot 10^{-10} x^6 - 2.44997581398416 \cdot 10^{-7} x^5 \\
& + 1.685493836491650 \cdot 10^{-4} x^4 - 6.12079611169612 \cdot 10^{-2} x^3 \\
& + 1.23619400253085 \cdot 10 x^2 - 1.32103052840034 \cdot 10^3 x \\
& + 5.8555709097502 \cdot 10^4
\end{aligned}
\qquad (3.1.3)
$$

$$
\begin{aligned}
v_{sat}^1 = {} & 1.18530523547986 \cdot 10^{-12} x^6 - 1.96308206488351 \cdot 10^{-9} x^5 \\
& + 1.35826188392364 \cdot 10^{-6} x^4 - 5.02771691419784 \cdot 10^{-4} x^3 \\
& + 1.05067895882963 \cdot 10^{-1} x^2 - 1.17618524656597 \cdot 10 x \\
& + 5.51575489976189 \cdot 10^2
\end{aligned}
\qquad (3.1.4)
$$

$$
\begin{aligned}
v_{sat}^2 = {} & 4.56273757620492 \cdot 10^{-14} x^6 - 9.09362775415188 \cdot 10^{-11} x^5 \\
& + 7.56366606191687 \cdot 10^{-8} x^4 - 3.36371424236542 \cdot 10^{-5} x^3 \\
& + 8.44466288226083 \cdot 10^{-3} x^2 - 1.13624863010097 x \\
& + 6.41270714320972 \cdot 10
\end{aligned}
\qquad (3.1.5)
$$

3.2 Συντελεστές και σταθερές του αέριου μίγματος R152a/R125/R32 (48/18/34)

Για το τριμερές μίγμα R152a / R125 / R32 (48/18/34 κατά μάζα) δεν υπάρχουν προτεινόμενες τιμές για τις περισσότερες σταθερές, λόγω του ότι το μίγμα είναι αρκετά καινούριο και δεν έχει μελετηθεί ακόμα.

Για το R152a δεν βρέθηκαν τιμές για τις σταθερές της Peng-Robinson. Έτσι θεωρήθηκαν μηδενικές τιμές για τα $\alpha_{2,i}$ και $\alpha_{3,i}$, δηλαδή μηδενικοί συντελεστές για τους παράγοντες της δεύτερης και τρίτης δύναμης και χρησιμοποιήθηκε μια προγενέστερη μορφή της Peng-Robinson, η οποία επιτρέπει τον υπολογισμό των συντελεστών $\alpha_{c,i}$, $\alpha_{1,i}$ και b από τις σχέσεις [5] :

$$a_{c,i} = 0.457235 \frac{R^2 T_c^2}{p_c} \tag{3.2.1}$$

$$a_{1,i} = 0.37464 + 1.54226\omega + 0.26992\omega^2 \tag{3.2.2}$$

$$b = 0.077796 \frac{RT_c}{p_c} \tag{3.2.3}$$

Για να μετρηθεί η επίδραση αυτή της απλοποίησης στα τελικά αποτελέσματα, συγκρίθηκαν για 480 σημεία εντός της υπέρθερμης περιοχής οι τιμές που υπολογίζονται για το R407b με και χωρίς αυτήν την απλούστευση. Τα σχετικά σφάλματα κυμάνθηκαν στο 0.020-0.111% για όλες τις υπολογιζόμενες μεταβλητές, γεγονός που επιτρέπει τη χρήση των σχέσεων 3.2.1, 3.2.2 και 3.2.3 για το νέο αέριο χωρίς απώλεια ακρίβειας.

Οι κρίσιμες τιμές θερμοκρασίας, πίεσης και ειδικού όγκου των συστατικών δίνονται [3], [10]:

Πίνακας 3.2.1 *Κρίσιμες και άλλες σταθερές του R152a / R125 / R32*

Συστατικό	Κρίσιμη T_c (K)	Κρίσιμη p_c (kPa)	Κρίσιμος V_c (cm^3/mol)	Μ. βάρος (kg/kmol)	Συντελεστής εκκεντρότητας	Σύσταση (κατά μάζα)
R152a	386,44	4520,0	179,4837	66,050	0,252851	0,48
R125	339,45	3630,6	211,40	120,0215	0,294334	0,18
R32	351,55	5830,0	123,8096	52,0235	0,267817	0,34

Όπως και στο R407b οι τιμές s_o και u_o του παρακάτω πίνακα επιλέγονται έτσι ώστε να ταυτίζονται οι υπολογιζόμενες από τον κώδικα τιμές της εντροπίας και της ενθαλπίας με τις αντίστοιχες που δίνει το πρόγραμμα REFPROP για το σημείο p_o=251,325kPa, T_o=273,15K. Οι συντελεστές A, B, C, D σύμφωνα με την έκφραση 2.2.18 του ιδανικού c_p παρατίθενται στον ίδιο πίνακα [2], [10].

Πίνακας 3.2.2: *Σταθερές αναφοράς και ιδανικού c_p του R152a / R125 / R32*

	R152a / R125 / R32
p_o (kPa)	251,325
T_o (K)	273,15
s_o (kJ/kg K)	2,131
u_o (kJ/kg)	455,30
A	$4{,}82584348 \times 10^{-1}$
B	$6{,}29912660 \times 10^{-4}$
C	$4{,}04879840 \times 10^{-6}$
D	$-4{,}24878898 \times 10^{-9}$

Πληροφορίες για τους συντελεστές αλληλεπίδρασης k_{ij}, k_{123} και l_{ij} του νέου μίγματος δεν υπάρχουν, γι' αυτό θεωρήθηκαν μηδενικοί. Για την απλοποίηση αυτή ακολουθήθηκε η ίδια διαδικασία σύγκρισης των τιμών με και χωρίς τη χρήση των διορθωτικών συντελεστών για το R407b. Η μελέτη των αποκλίσεων έδειξε σχετικά σφάλματα επιπέδου 0.001-0.015%, οπότε η υπόθεση μηδενικών συντελεστών δεν επηρεάζει ουσιαστικά τα αποτελέσματα και μπορούν να αμεληθούν.

Οι πολυωνυμικές εξισώσεις που μοντελοποιήθηκαν στο Excell για τις καμπύλες κορεσμένου αερίου p_{sat} και v_{sat} με βάση το πρόγραμμα REFPROP φαίνονται παρακάτω. Για τον όγκο χρησιμοποιήθηκαν και γι' αυτό το μίγμα δύο εξισώσεις $6^{ου}$ βαθμού για μεγαλύτερη ακρίβεια. Το ανώτατο σφάλμα της μοντελοποίησης είναι 0.019% για το v_{sat} και 0.384% για τη p_{sat}.

$$\begin{aligned} p_{sat} = {} & 5.69458483406365 \cdot 10^{-10} x^6 - 9.74950338883352 \cdot 10^{-7} x^5 \\ & + 6.97039451880329 \cdot 10^{-4} x^4 - 2.65021689221286 \cdot 10^{-1} x^3 \\ & + 5.647145802326 \cdot 10 x^2 - 6.39650290891730 \cdot 10^{3} x \\ & + 3.01055282943808 \cdot 10^{5} \end{aligned} \qquad (3.2.4)$$

$$\begin{aligned} v_{sat}^{1} = {} & 2.63572780048964 \cdot 10^{-12} x^6 - 4.48159674225455 \cdot 10^{-9} x^5 \\ & + 3.18157053817254 \cdot 10^{-6} x^4 - 1.20756541039049 \cdot 10^{-3} x^3 \\ & + 2.58571759571226 \cdot 10^{-1} x^2 - 2.963517413720270 \cdot 10 x \\ & + 1.42150529203835 \cdot 10^{3} \end{aligned} \qquad (3.2.5)$$

$$\begin{aligned} v_{sat}^{2} = {} & 1.740442113041080 \cdot 10^{-14} x^6 - 4.21168350304823 \cdot 10^{-11} x^5 \\ & + 4.15785301486820 \cdot 10^{-8} x^4 - 2.15890107344439 \cdot 10^{-5} x^3 \\ & + 6.25028354696890 \cdot 10^{-3} x^2 - 9.60589224331175 \cdot 10^{-1} x \\ & + 6.14566526436002 \cdot 10 \end{aligned} \qquad (3.2.6)$$

Κεφάλαιο 4ο :

4.1 Παρουσίαση και αξιολόγηση αποτελεσμάτων

Σε αυτό το σημείο, όπου έχει διατυπωθεί το μαθηματικό μοντέλο της Peng-Robinson, έχει περιγραφεί ο κώδικας και έχουν δοθεί οι σχετικές παράμετροι των δύο αερίων, μπορούν να παρατεθούν τα αποτελέσματα που δίνει το πρόγραμμα σε java και να συγκριθούν με τα διεθνώς αποδεκτά του REFPROP της NIST.

Τα εξαγόμενα μεγέθη του προγράμματος παρουσιάζονται σε διαγράμματα συναρτήσει της θερμοκρασίας και της πίεσης, θεωρώντας τις ως τις πλέον καθοριστικές ιδιότητες για τις εφαρμογές. Τα σχετικά διαγράμματα δίνονται στο Παράρτημα Α.

Ο έλεγχος των τιμών των ιδιοτήτων βασίστηκε στη σύγκριση 480 κατανεμημένων σημείων που ορίζονται από τη θερμοκρασία και τον όγκο για όλες τις εξαγόμενες μεταβλητές του προγράμματος σε java με τις τιμές που δίνει το REFPROP γι' αυτές. Για την εκτίμηση των αποκλίσεων υπολογίστηκε το σχετικό σφάλμα (η απόλυτη διαφορά των τιμών προς αυτή που δίνει το REFPROP).

Για τον προσδιορισμό τους χρησιμοποιήθηκαν 24 διαφορετικές θερμοκρασίες του συνολικού εύρους που διερευνήθηκε (240,15K-481,05K), εκκινώντας από τους 243,15K (-30^{o}C) , με βήμα 10K μέχρι τους 473,15K (200 ^{o}C). Για κάθε τιμή θερμοκρασίας επιλέχθηκαν 20 ισαπέχουσες τιμές όγκου του συνολικού εύρους που αντιστοιχεί στην εκάστοτε θερμοκρασία. Αυτό συμβαίνει γιατί σε θερμοκρασίες χαμηλότερης της κρίσιμης, για μικρούς όγκους το μίγμα βρίσκεται στη διφασική περιοχή και το μοντέλο της Peng-Robinson που χρησιμοποιήθηκε δεν έχει σαφώς εφαρμογή. Η τιμή όγκου που αντιστοιχεί στη γραμμή κορεσμένου αερίου ορίζει την πρώτη τιμή όγκου ελέγχου. Το συνολικό εύρος καθορίζεται ως $v_{sat} - 0,6$ m^3/kg και σε αυτό το διάστημα ορίζονται τα 20 ισαπέχοντα σημεία. Για τις τιμές της θερμοκρασίας όπου αντιστοιχεί v_{sat} μικρότερο του 0,006 m^3/kg , η οποία είναι και η μικρότερη τιμή όγκου για την οποία ορίστηκε να λειτουργεί το πρόγραμμα της java, το εύρος όγκου ορίζεται ως 0,006 m^3/kg – 0,6 m^3/kg. Οι πληροφορίες για τα σημεία κορεσμού αντλήθηκαν από το πρόγραμμα REFPROP [2] και όπως αναφέρθηκε αφού μοντελοποιήθηκαν από πολυωνυμικές εξισώσεις 6^{ου} βαθμού ενσωματώθηκαν στον κώδικα.

Τα κρίσιμα μεγέθη θερμοκρασίας, πίεσης και όγκου που υπολογίστηκαν από το πρόγραμμα σύμφωνα με την προτεινόμενη μέθοδο για τη καταστατική εξίσωση Lee-Kesler και για τη Peng-Robinson συγκρίθηκαν με αυτά που δίνει το REFPROP. Ο έλεγχος έδειξε και για τα δύο αέρια ότι η προτεινόμενη μέθοδος για τη Peng-Robinson παρουσιάζει μη αποδεκτά αποτελέσματα ενώ αντίθετα η προτεινόμενη για τη Lee-Kesler εμφανίζει σφάλματα κάτω του 2,5%. Αξίζει να σημειωθεί ότι στο εγχειρίδιο του προγράμματος REFPROP [2] αναφέρεται ότι οι υπολογιζόμενες κρίσιμες ιδιότητες ενδέχεται να παρουσιάζουν σημαντική απόκλιση από τις πραγματικές. Στην παρούσα εργασία αποδεχόμαστε τις τιμές του REFPROP.

Τα αποτελέσματα για το μίγμα R407b ήταν πολύ θετικά καθώς τα σχετικά σφάλματα όλων των ιδιοτήτων ήταν για κάθε συνδυασμό θερμοκρασίας και όγκου της τάξης του 1% ή και μικρότερης. Μοναδική εξαίρεση αποτέλεσαν τα σημεία που αντιστοιχούν σε ακραία χαμηλούς όγκους, είτε κοντά δηλαδή στη γραμμή κορεσμένου αερίου, είτε κοντά στην τιμή $0{,}006 m^3/kg$, η οποία είναι πολύ κοντινή της κρίσιμης $v_c = 0{,}001880866\, m^3/kg$ (τιμή REFPROP). Σε αυτές τις περιοχές το σφάλμα κορυφωνόταν ανάλογα με το μέγεθος στο 1-18%, αλλά για όλες τις μεταβλητές η απόκλιση αυτή ήταν σημειακή και το σφάλμα έπεφτε ραγδαία στις επόμενες τιμές όγκου. Οι υπολογισμοί αυτοί επιβεβαιώνουν τον ισχυρισμό ότι το μοντέλο της Peng-Robinson προσεγγίζει ικανοποιητικά τη συμπεριφορά του μίγματος στην υπέρθερμη περιοχή, γεγονός αναμενόμενο καθώς είχε εφαρμοστεί και παλαιότερα το μοντέλο αυτό για το συγκεκριμένο μίγμα στο εργαστήριο.

Τα αποτελέσματα που προέκυψαν για το τριμερές μίγμα R152a / R125 / R32 δεν ήταν για όλες τις ιδιότητες ικανοποιητικά. Οι αποκλίσεις των μεγεθών p, s, h, $k_{p,T}$, $k_{p,v}$, $k_{T,v}$, γ, Z και α διατηρήθηκαν σε χαμηλά επίπεδα με σχετικά σφάλματα της τάξης του 1% ή και μικρότερης με σημειακή εξαίρεση τους ακραία χαμηλούς όγκους (η κρίσιμη τιμή όγκου γι' αυτό το μίγμα είναι $v_c = 0{,}002477639\, m^3/kg$ – τιμή REFPROP). Σε αυτές τις περιοχές το σφάλμα κορυφωνόταν ανάλογα με το μέγεθος στο 1-8% και όπως στο R407b ακολουθούσε απότομη πτώση του από την επόμενη τιμή όγκου. Διαφορετική συμπεριφορά εμφάνισαν οι ειδικές θερμοχωρητικότητες c_v και c_p για τις οποίες παρουσιάστηκαν στους ακραία μικρούς όγκους πολύ υψηλά σφάλματα, αγγίζοντας το 19% για ορισμένες θερμοκρασίες ενώ ταυτόχρονα η πτώση του σφάλματος για αυξανόμενο όγκο είχε μικρότερο ρυθμό. Συνέπεια αυτού ήταν η διατήρηση των σφαλμάτων σε υψηλά επίπεδα (2-8%), ιδιαίτερα για τις μικρές θερμοκρασίες, κρίνοντας τη χρήση του προγράμματος απαγορευτική για τις συγκεκριμένες μεταβλητές και για τη συγκεκριμένη περιοχή χαμηλών θερμοκρασιών και όγκων.

Ακόλουθα παρουσιάζεται η ανάλυση σφαλμάτων για το κάθε μίγμα και όπου κρίνεται απαραίτητο, τα σχετικά διαγράμματα για την άμεση εποπτική εύρεση της απόκλισης του επιθυμητού συνδυασμού θερμοκρασίας και όγκου.

4.2 Ανάλυση και σχολιασμός των σχετικών σφαλμάτων του R407b

Τα κρίσιμα μεγέθη, όπως υπολογίστηκαν από τις δύο προτεινόμενες μεθόδους παρουσιάζονται στον παρακάτω πίνακα μαζί με τις συγκρινόμενες τιμές του προγράμματος REFPROP [2] και τα σχετικά σφάλματα. Από τον πίνακα φαίνεται καθαρά ότι η μοναδική μέθοδος με αποδεκτές αποκλίσεις είναι η προτεινόμενη για την καταστατική εξίσωση Lee-Kesler. Στον ίδιο πίνακα παρατίθενται και το μοριακό βάρος, η σταθερά του αερίου και ο συντελεστής εκκεντρότητας μαζί με τα σφάλματα τους.

Πίνακας 4.2.1:*Σφάλματα κρίσιμων μεγεθών του R407b*

	REFPROP	**Lee-Kesler**	**Σφάλμα % Lee-Kesler**	**Peng-Robinson**	**Σφάλμα % Peng-Robinson**
T_{cm} (K)	348,12	348,01	0,032	347,15	0,279
p_{cm}(kPa)	4130,2	4036,0	2,281	4007,4	2,973
v_{cm}(m^3/kg)	0,00188087	0,00184905	1,691	0,00174728	7,102
M(kg/kmol)	102,94	-	-	102,937	0,003
R(kJ/kgK)	0,080769838	-	-	0,080772463	0,003
ω_m	-	-	-	0,293470	-

Τα αποτελέσματα που προέκυψαν για το R407b, όπως αναφέρθηκε ήταν ιδιαίτερα ικανοποιητικά για όλα τα μεγέθη με αξιοσημείωτες αποκλίσεις μόνο για πολύ μικρούς όγκους.

Οι τιμές σφαλμάτων που παρουσιάστηκαν για τις ιδιότητες p, s, h, και Z κυμάνθηκαν γενικά σε πολύ χαμηλά ποσοστά κάτω του 2%, εμφανίζοντας πτωτική τάση για αύξοντα όγκο. Η άνοδος της θερμοκρασίας αρχικά είχε αύξουσα επιρροή και στα σφάλματα μέχρι μια κεντρική θερμοκρασία που ποίκιλε ανάλογα με το μέγεθος (313,15-373,15K) ενώ στη συνέχεια ακολουθούσε πτωτική πορεία.

Τα σφάλματα των μεταβλητών $k_{p,T}$, $k_{p,v}$, $k_{T,v}$, γ και α εμφάνισαν πτωτική συμπεριφορά στην αύξηση των όγκων αλλά ανοδική στην αύξηση της θερμοκρασίας. Σημειώθηκε πτώση τους στη μετάβαση από τους 313,15K στους 323,15K αλλά μετά συνεχίστηκε η ανοδική πορεία τους. Αξίζει να αναφερθεί ότι τα σφάλματα του ισεντροπικού συντελεστή $k_{T,v}$ δεν ξεπέρασαν σε κανένα σημείο το 1%.

Οι ειδικές θερμοχωρητικότητες c_v, c_p σε αύξηση του όγκου δεν εμφάνισαν πάντοτε πτωτική πορεία σφάλματος. Στην περιοχή 263,15K - 433,15K παρατηρήθηκε αύξηση του για τους υψηλούς όγκους με κορύφωση στους 353,15K όπου το επίπεδο διατηρήθηκε σταθερά στο 1-1,5%. Πέρα από την εμφάνιση αυτού του φαινομένου τα υψηλότερα ποσοστά σφάλματος εμφανίστηκαν και σε αυτά τα μεγέθη είτε κοντά στους όγκους κορεσμού, είτε στον οριακό της μελέτης όγκο 0,006 m^3/kg. Η τάση που ακολούθησαν οι αποκλίσεις των μικρών όγκων όσον αφορά τη θερμοκρασία ήταν ανοδική μέχρι τους 333,15K και μετέπειτα καθοδική. Το επίπεδο τους ήταν ιδιαίτερα

υψηλό σε σύγκριση με τις υπόλοιπες ιδιότητες, της τάξης 10-18%, γεγονός που συνιστά απαγορευτική τη χρήση της Peng-Robinson για τον υπολογισμό των c_v, c_p σε συγκεκριμένο εύρος όγκου. Το φαινόμενο όμως εμφάνισε έντονη πόλωση με αποτέλεσμα το σφάλμα των υπολογισμών να πέσει γρήγορα κάτω του 2%. Με κριτήριο απόρριψης το 2% ως ελάχιστη επιτρεπόμενη τιμή όγκου για θερμοκρασίες 243,15-273,15K βρέθηκε η $v - v_{sat} = 0,08 m^3/kg$, για θερμοκρασίες 273,15-323,15K η $v - v_{sat} = 0,06 m^3/kg$, ενώ για μεγαλύτερες θερμοκρασίες ο ελάχιστος όγκος παίρνει τιμές $v - 0,006 = 0,035 \Rightarrow v = 0,041 m^3/kg$ ή και ακόμα μικρότερες.

Η έντονη παρουσία υψηλών αποκλίσεων που παρατηρήθηκε για μικρούς όγκους και κεντρικές θερμοκρασίες σε όλα τα μεγέθη αντικατοπτρίζει την επαλληλία των σφαλμάτων που επιφέρει η επίδραση της μετάβασης στη διφασική περιοχή και η επίδραση της προσέγγισης των τιμών του κρίσιμου σημείου. Σε αυτές τις περιοχές, τις πλέον δύσκολες για τον προσδιορισμό των θερμοφυσικών μεγεθών το μοντέλο της Peng-Robinson τελικά ανταποκρίνεται με επιτυχία για τις μεταβλητές p, s, h, γ, $k_{p,T}$, $k_{p,v}$, $k_{T,v}$, Z και α, ενώ για τα c_v και c_p συνιστάται προσοχή.

Παρακάτω παρουσιάζονται οι μέσες και οι μέγιστες τιμές σφαλμάτων καθώς και ο όγκος και η θερμοκρασία, όπου εμφανίζονται οι μέγιστες αυτές τιμές.

Πίνακας 4.2.2: *Μέσα και μέγιστα σφάλματα θερμοδυναμικών μεγεθών του R407b*

Μέγεθος	**Μέσο σφάλμα%**	**Μέγιστο σφάλμα%**	**Τ μέγιστου σφάλματος(K)**	**v μέγιστου σφάλματος(m^3/kg)**
p	0,156	1,653	373,15	0,0060
s	0,104	1,278	333,15	0,0060
h	0,140	1,089	323,15	0,0075 (sat)
c_v	1,237	14,606	333,15	0,0060
c_p	1,221	18,838	323,15	0,0075 (sat)
k_{pv}	0,324	4,135	473,15	0,0060
k_{tv}	0,158	0,844	473,15	0,0060
k_{pt}	0,148	1,120	333,15	0,0060
γ	0,350	6,054	323,15	0,0075 (sat)
Z	0,156	1,655	373,15	0,0060
α	0,161	2,068	473,15	0,0060

4.3 Ανάλυση και σχολιασμός των σχετικών σφαλμάτων του R152/R125/R32 (48/18/34)

Τα κρίσιμα μεγέθη, όπως προσδιορίστηκαν και με τις δύο προτεινόμενες μεθόδους παρουσιάζονται στον παρακάτω πίνακα μαζί με τις συγκρινόμενες τιμές του προγράμματος REFPROP [2] και τα σχετικά σφάλματα. Από τον πίνακα φαίνεται καθαρά ότι η μοναδική μέθοδος με αποδεκτές αποκλίσεις είναι η προτεινόμενη για τη καταστατική εξίσωση Lee-Kesler, όπως και για το R407b. Στον ίδιο πίνακα παρατίθενται και το υπολογιζόμενο μοριακό βάρος, η σταθερά του αερίου και ο συντελεστής εκκεντρότητας μαζί με τα σφάλματα τους.

Πίνακας 4.3.1: *Σφάλματα κρίσιμων μεγεθών του R152a/R125 /R32*

	REFPROP	**Lee-Kesler**	**Σφάλμα % Lee-Kesler**	**Peng-Robinson**	**Σφάλμα % Peng-Robinson**
T_{cm} (K)	367,93	366,23	0,462	365,82	0,573
p_{cm}(kPa)	5095,4	5174,7	1,556	4844,0	4,934
v_{cm}(m^3/kg)	0,00247764	0,00241411	2,564	0,00298954	20,661
M(kg/kmol)	65,35	-	-	65,355	0,008
R(kJ/kgK)	0,127229872	-	-	0,127219877	0,008
ω_m	-	-	-	0,263308	-

Το νέο τριμερές μίγμα εμφάνισε γενικά αποδεκτή συμπεριφορά για τις ιδιότητες p, s, h, $k_{p,T}$, $k_{p,v}$, $k_{T,v}$, γ, Z και α σε αντιδιαστολή με τις ειδικές θερμοχωρητικότητες c_v, c_p οι οποίες προβλημάτισαν με την απόκριση τους σε ευρεία περιοχή όγκων και θερμοκρασιών.

Οι τιμές σφαλμάτων που παρουσιάστηκαν για τα μεγέθη p, s, h, $k_{p,T}$ και Z κυμάνθηκαν σε πολύ χαμηλά ποσοστά κάτω του 2% σε όλο το εύρος τιμών θερμοκρασίας και όγκου. Αύξηση του όγκου επέφερε γενικά πτώση του σφάλματος ενώ αύξηση της θερμοκρασίας επέφερε αρχικά άνοδο του μέχρι μια κεντρική θερμοκρασία που ποίκιλε ανάλογα με τη μεταβλητή (293,15-353,15K) ενώ στη συνέχεια οδηγούσε σε πτώση του.

Οι αποκλίσεις των ιδιοτήτων $k_{p,v}$, $k_{T,v}$, γ και α εμφάνισαν καθοδική τάση στην αύξηση των όγκων αλλά ανοδική στην αύξηση της θερμοκρασίας. Σημειώθηκε πτώση τους σε κάποια κεντρική τιμή θερμοκρασίας μεταξύ των θερμοκρασιών 283,15K και 353,15K ανάλογα με τη μεταβλητή, αλλά μετά συνεχίστηκε ανοδική πορεία σφαλμάτων.

Οι ειδικές θερμοχωρητικότητες c_v, c_p δεν εμφάνισαν ανοδική τάση σφάλματος για αύξηση του όγκου σε καμία περιοχή θερμοκρασίας σε αντίθεση με το R407b. Τα υψηλότερα ποσοστά εμφανίστηκαν και σε αυτά τα μεγέθη είτε κοντά στους όγκους κορεσμού, είτε στον οριακό της μελέτης όγκο 0,006 m^3/kg. Η πορεία που ακολούθησαν τα σφάλματα ως προς τη θερμοκρασία ήταν ανοδική μέχρι τους

343,15K και μετέπειτα καθοδική. Το επίπεδο τους εμφανίστηκε ιδιαίτερα υψηλό σε σύγκριση με τις υπόλοιπες ιδιότητες, της τάξης 10-20% και με μεγάλη διασπορά ιδιαίτερα στις χαμηλές θερμοκρασίες. Αυτό καθιστά απαγορευτική τη χρήση της Peng-Robinson για τον υπολογισμό των c_v, c_p σε μεγάλο εύρος περιοχών θερμοκρασίας και όγκου.

Οι μεγάλες αποκλίσεις που εμφανίστηκαν σε χαμηλούς όγκους και μεσαίες θερμοκρασίες, δείχνουν για άλλη μια φορά την αδυναμία προσδιορισμού των θερμοφυσικών μεγεθών κοντά στη διφασική περιοχή και στο κρίσιμο σημείο. Στην περίπτωση αυτού του αερίου μπορούμε να αποφανθούμε ότι για τις ιδιότητες p, s, h, $k_{p,T}$, $k_{p,v}$, $k_{T,v}$, γ, Z και α το μοντέλο της Peng-Robinson ανταποκρίνεται με μεγάλη ακρίβεια, ενώ για τις ειδικές θερμοχωρητικότητες c_v, c_p οι σημαντικές αποκλίσεις τους για μεγάλες περιοχές θερμοκρασίας και όγκου την καθιστούν ακατάλληλη για χρήση.

Ακολουθεί ο πίνακας μέσων και μέγιστων σφαλμάτων μαζί με τις θερμοκρασίες και τους όγκους όπου εμφανίζονται οι μέγιστες τιμές αυτών.

Πίνακας 4.3.2: *Μέσα και μέγιστα σφάλματα θερμοδυναμικών μεγεθών του R152a/R125/R32*

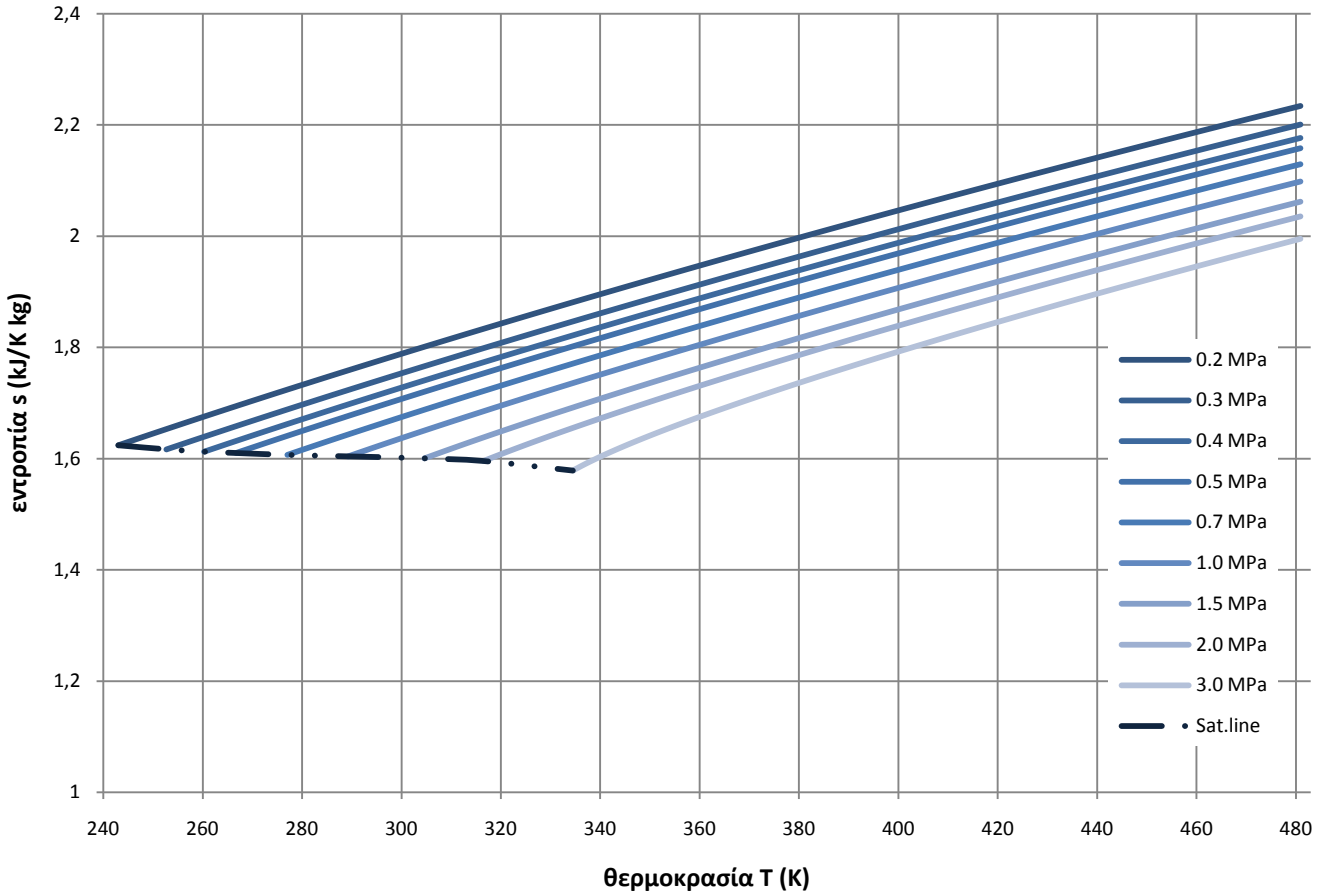
Μέγεθος	**Μέσο σφάλμα%**	**Μέγιστο σφάλμα%**	**Τ μέγιστου σφάλματος(K)**	**ν μέγιστου σφάλματος(m^3/kg)**
p	0,243	2,042	293,15	0,0395 (sat)
s	0,253	1,537	353,15	0,0070 (sat)
h	0,360	1,224	343,15	0,0095 (sat)
c_v	1,491	18,948	353,15	0,0070(sat)
c_p	1,722	19,510	343,15	0,0095 (sat)
k_{pv}	0,372	8,355	393,15	0,0060
k_{tv}	0,114	2,671	363,15	0,0060
k_{pt}	0,088	2,467	353,15	0,0070 (sat)
γ	0,485	7,125	363,15	0,0075 (sat)
Z	0,243	2,052	293,15	0,0395 (sat)
α	0,443	4,499	473,15	0,0060

Το επίπεδο σφαλμάτων και η υψηλή διασπορά τους που χαρακτηρίζει τα μεγέθη c_v, c_p, $k_{p,T}$, γ και α κατέστησαν αναγκαία την διερεύνηση των αποκλίσεων τους σε ευρύτερο πεδίο τιμών όγκου και θερμοκρασίας. Έτσι επιλέχθηκε ο υπολογισμός των σφαλμάτων ανά 1K και σε όλο το υπολογιζόμενο εύρος όγκου του προγράμματος (ανά 0.0005m^3/kg). Ο δεύτερος αυτός έλεγχος χρησιμοποίησε 27.152 τιμές έναντι 480 του πρώτου και οδήγησε στη δημιουργία των αντίστοιχων διαγραμμάτων. Με αυτόν τον τρόπο γίνεται δυνατός ο ακριβής προσδιορισμός του σφάλματος για τον επιθυμητό συνδυασμό θερμοκρασίας και όγκου. Αυτό διευκολύνει πολύ το χρήστη του προγράμματος, ο οποίος θέλει να γνωρίζει κατά πόσο μπορεί να

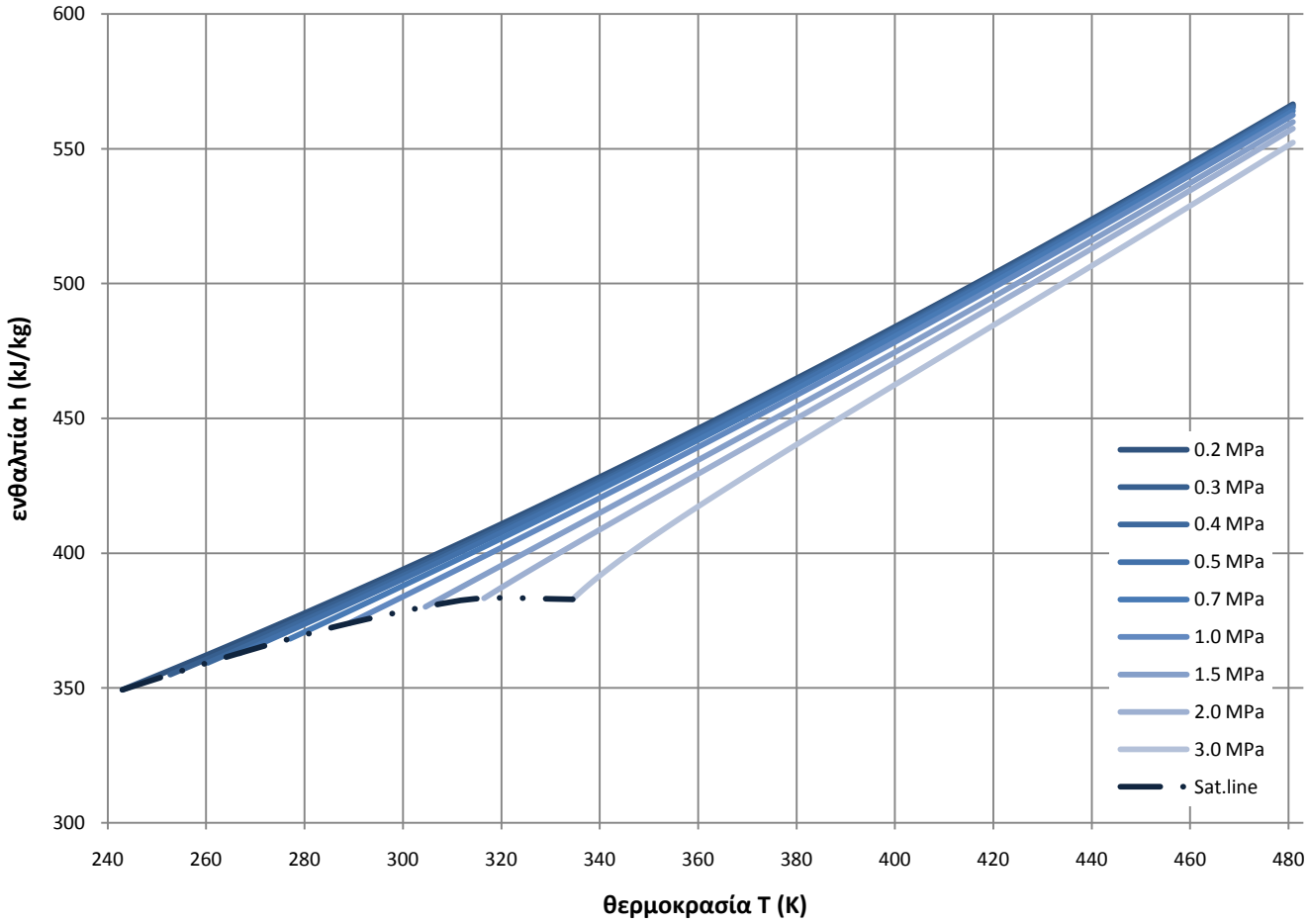
εμπιστεύεται τα αποτελέσματα για την περιοχή που τον ενδιαφέρει. Αξίζει να σημειωθεί ότι τα μέσα σφάλματα που σημειώθηκαν στον δεύτερο έλεγχο ήταν ελαφρά μικρότερα από αυτά του πρώτου. Αυτό οφείλεται στο γεγονός ότι τα μεγάλα σφάλματα των ακραίων τιμών, τα οποία περιλαμβάνονται και στους δύο ελέγχους είναι πολύ λιγότερα, οπότε έχουν όλο και λιγότερη επιρροή με την αύξηση του δείγματος. Στο Παράρτημα Β παρουσιάζονται τα προαναφερθέντα διαγράμματα.

ΠΑΡΑΡΤΗΜΑ Α: Διαγράμματα θερμοφυσικών μεγεθών

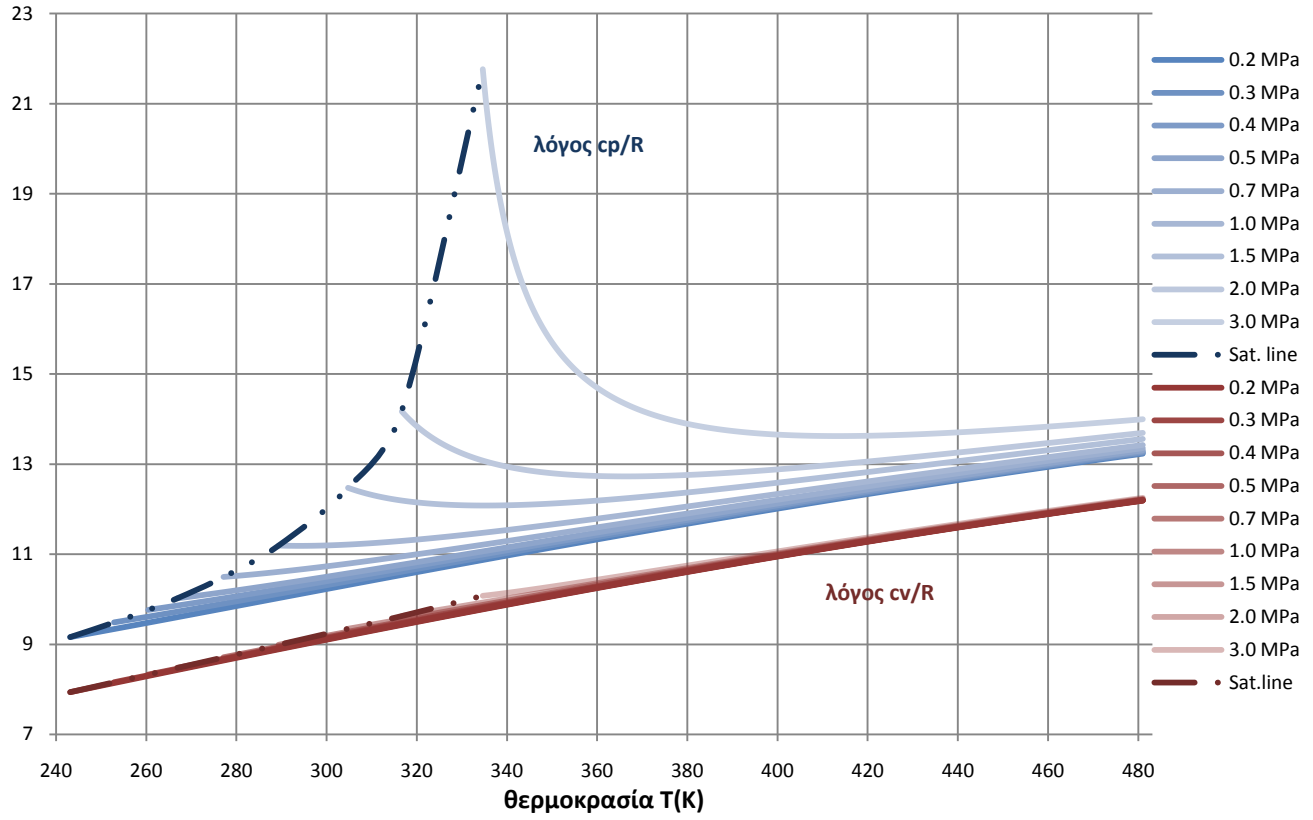
Σχήμα Α.1: *Διάγραμμα εντροπίας συναρτήσει της θερμοκρασίας και της πίεσης για το R407b*



Σχήμα Α.2: *Διάγραμμα ενθαλπίας συναρτήσει της θερμοκρασίας και της πίεσης για το R407b*

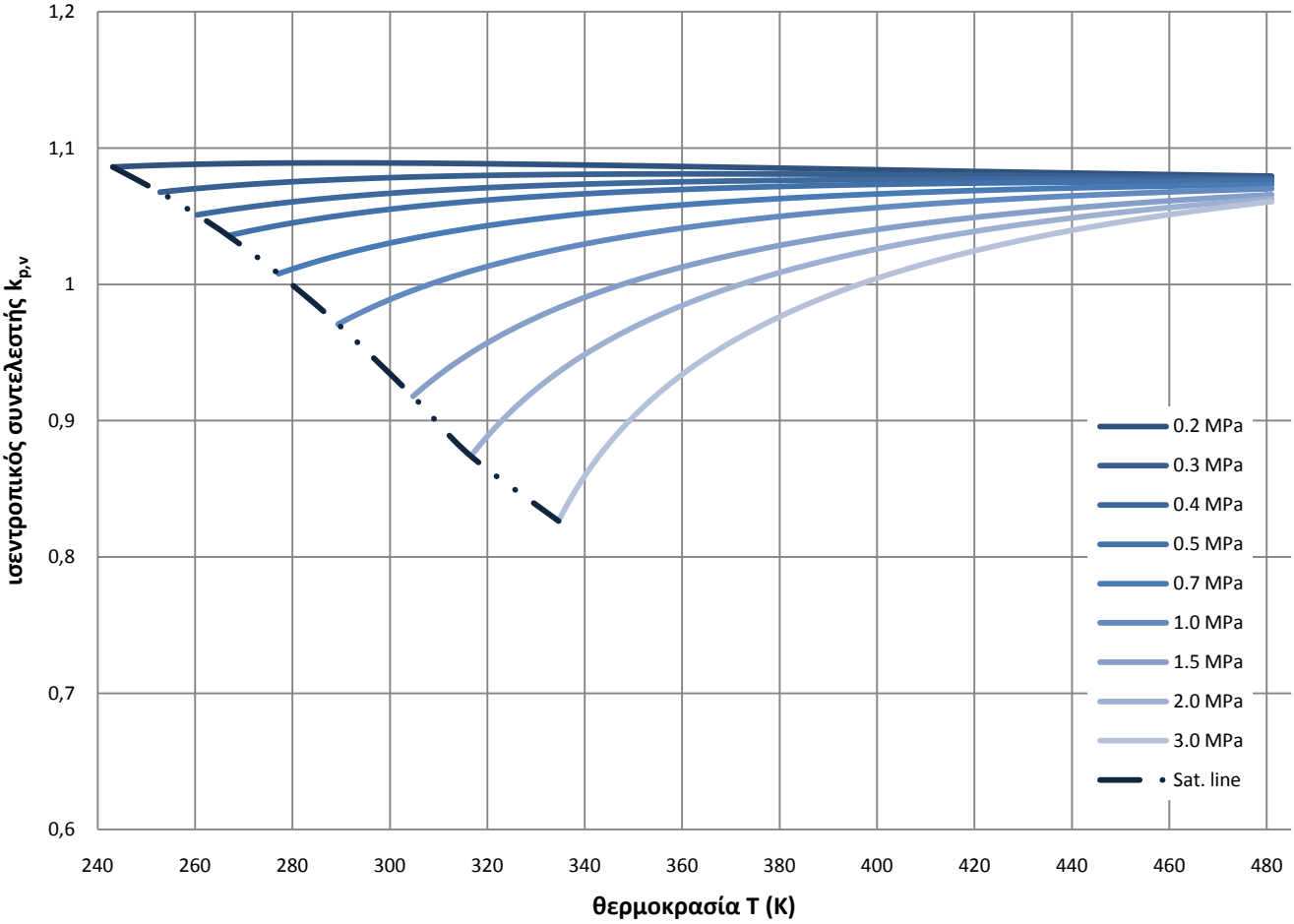


Σχήμα Α.3: *Διαγράμματα λόγων c_p / R και c_v / R συναρτήσει της θερμοκρασίας και της πίεσης για το R407b*

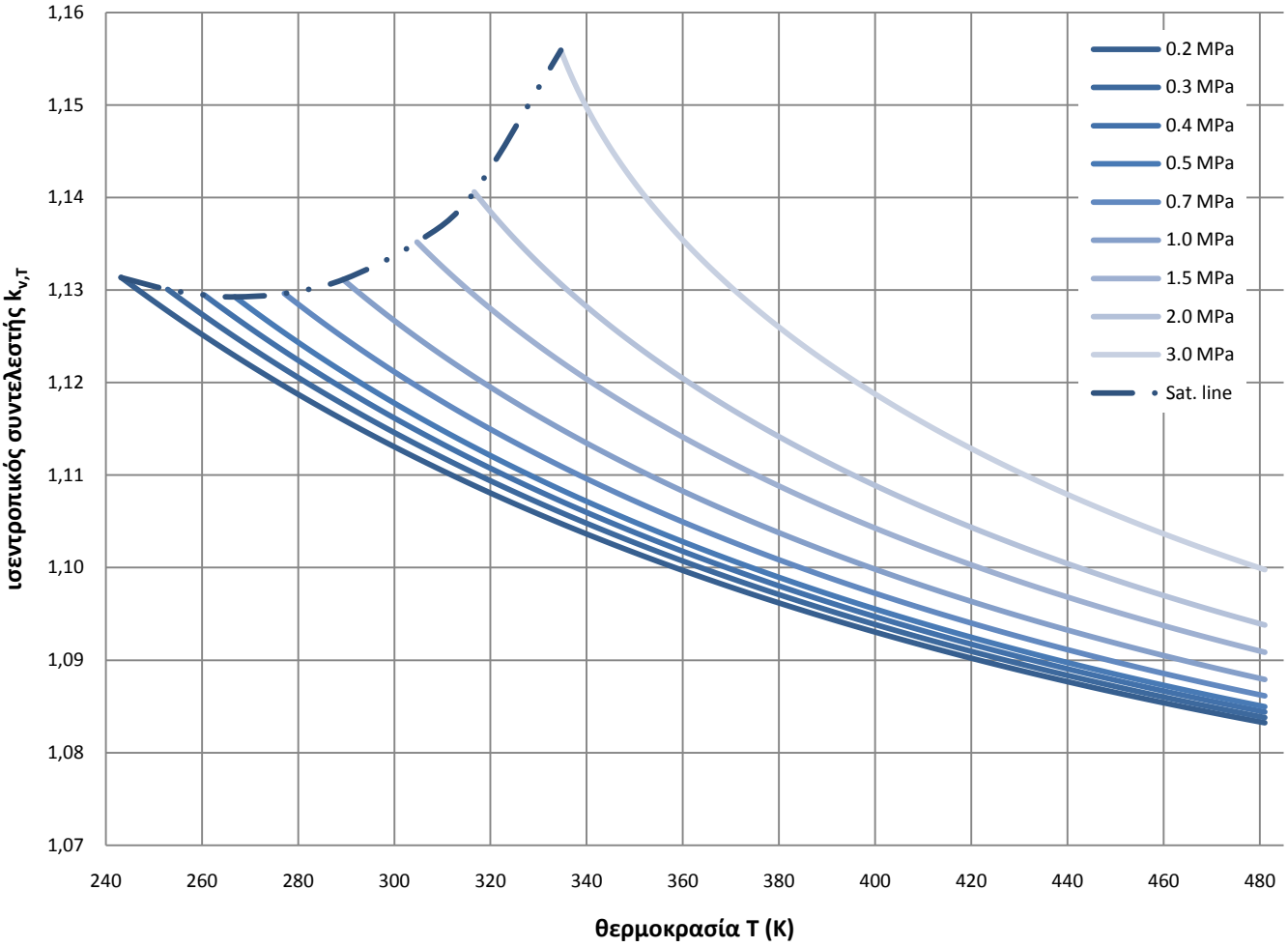


Τα ιδανικά αέρια εμφανίζουν διαφορά $\frac{c_p}{R} - \frac{c_v}{R} = 1$, σχέση που επαληθεύεται για το αέριο R407b σε πιέσεις μικρότερες των 0,2MPa, όπως φαίνεται στο σχήμα Α.3.

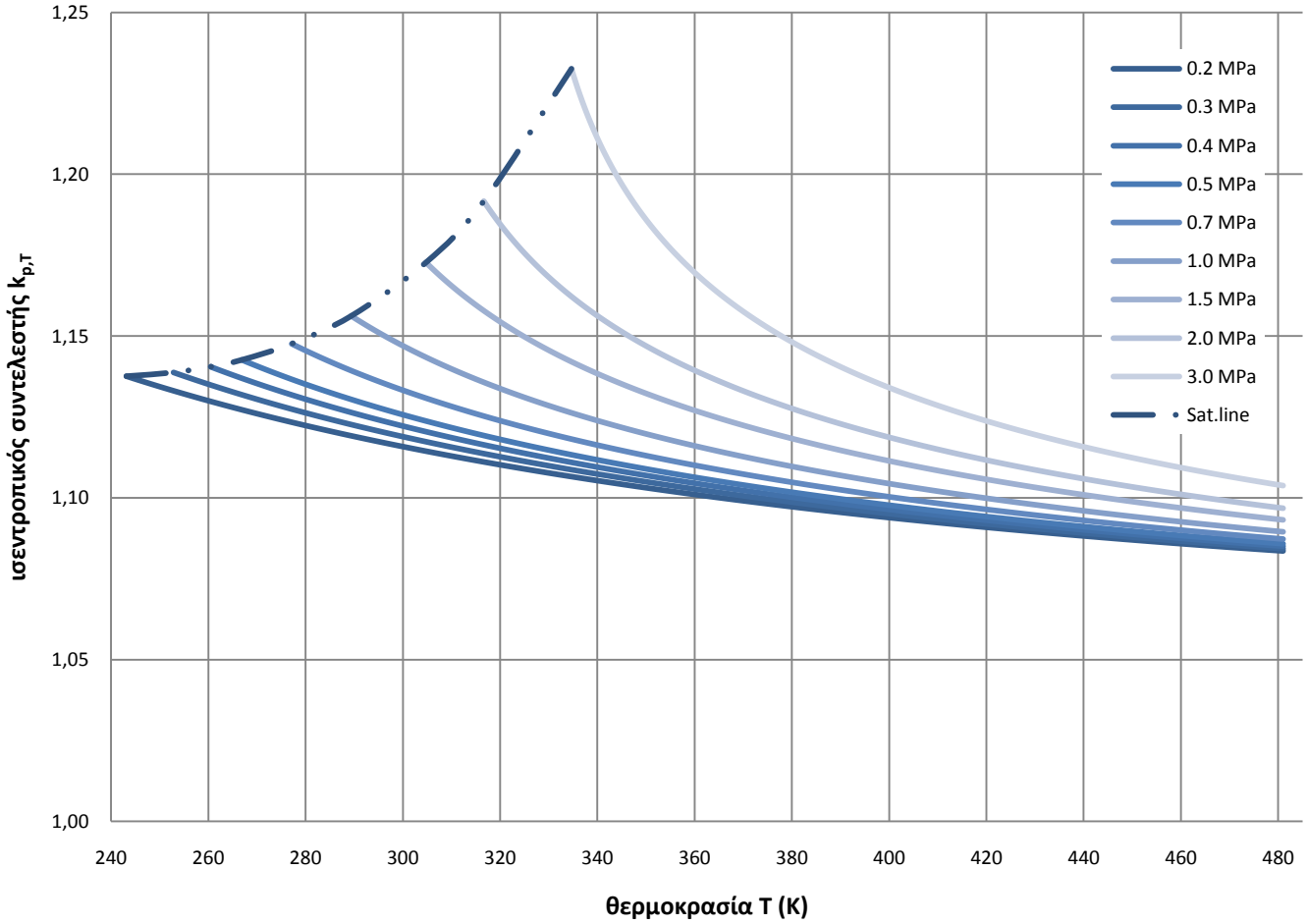
Σχήμα Α.4: *Διάγραμμα ισεντροπικού συντελεστή $k_{p,v}$ συναρτήσει της θερμοκρασίας και της πίεσης για το R407b*



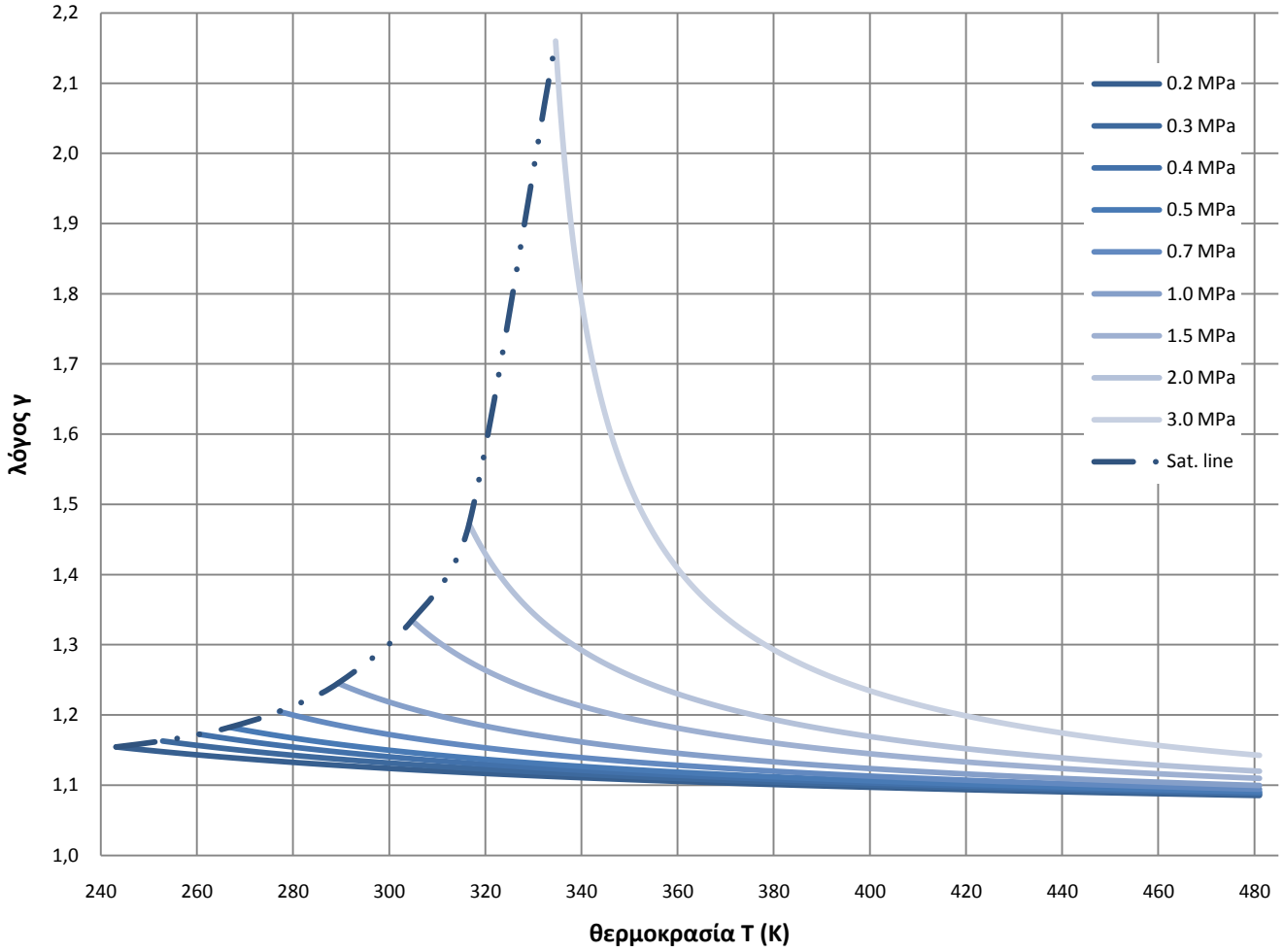
Σχήμα Α.5: *Διάγραμμα ισεντροπικού συντελεστή $k_{v,T}$ συναρτήσει της θερμοκρασίας και της πίεσης για το R407b*



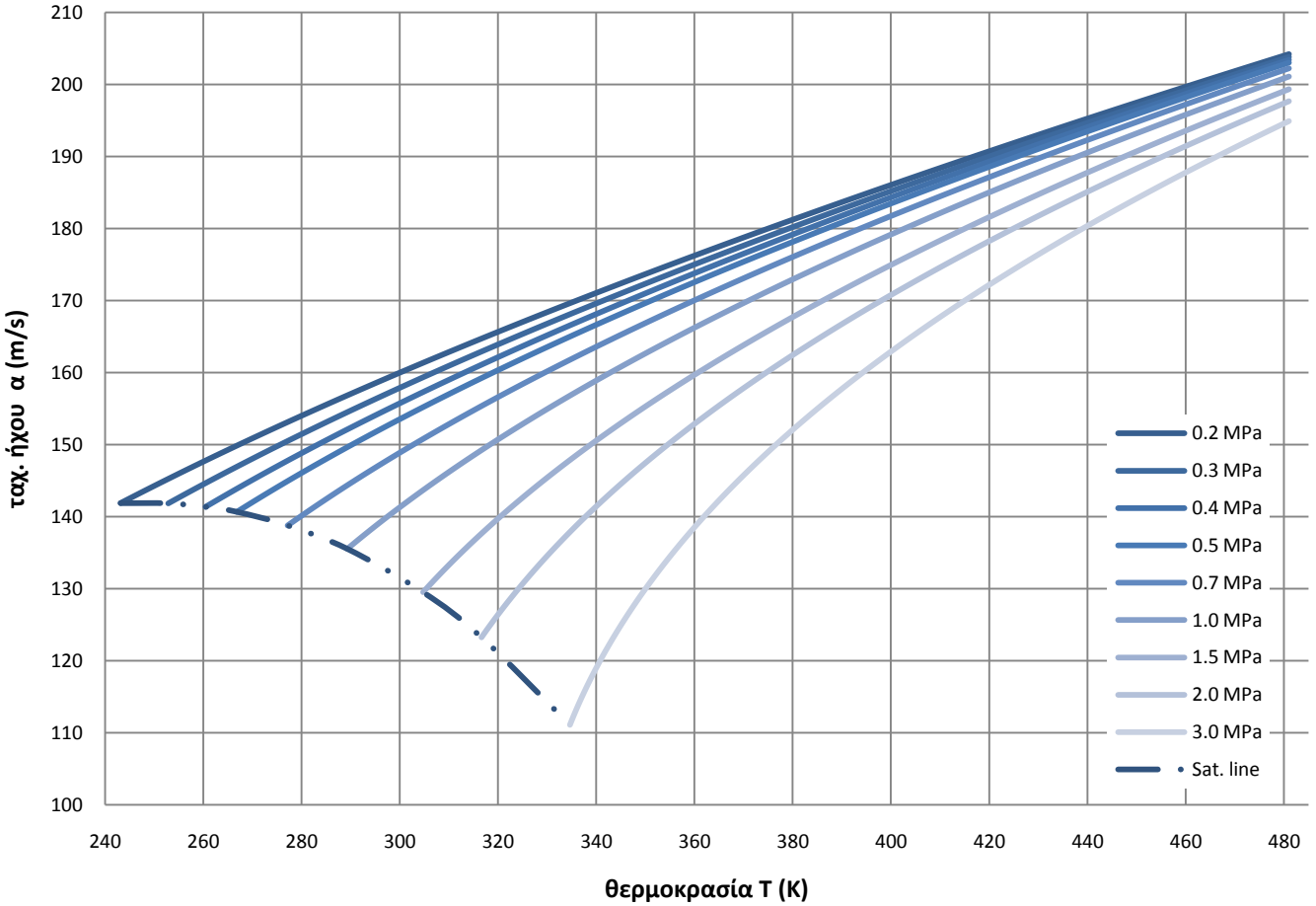
Σχήμα Α.6: *Διάγραμμα ισεντροπικού συντελεστή $k_{p,T}$ συναρτήσει της θερμοκρασίας και της πίεσης για το R407b*



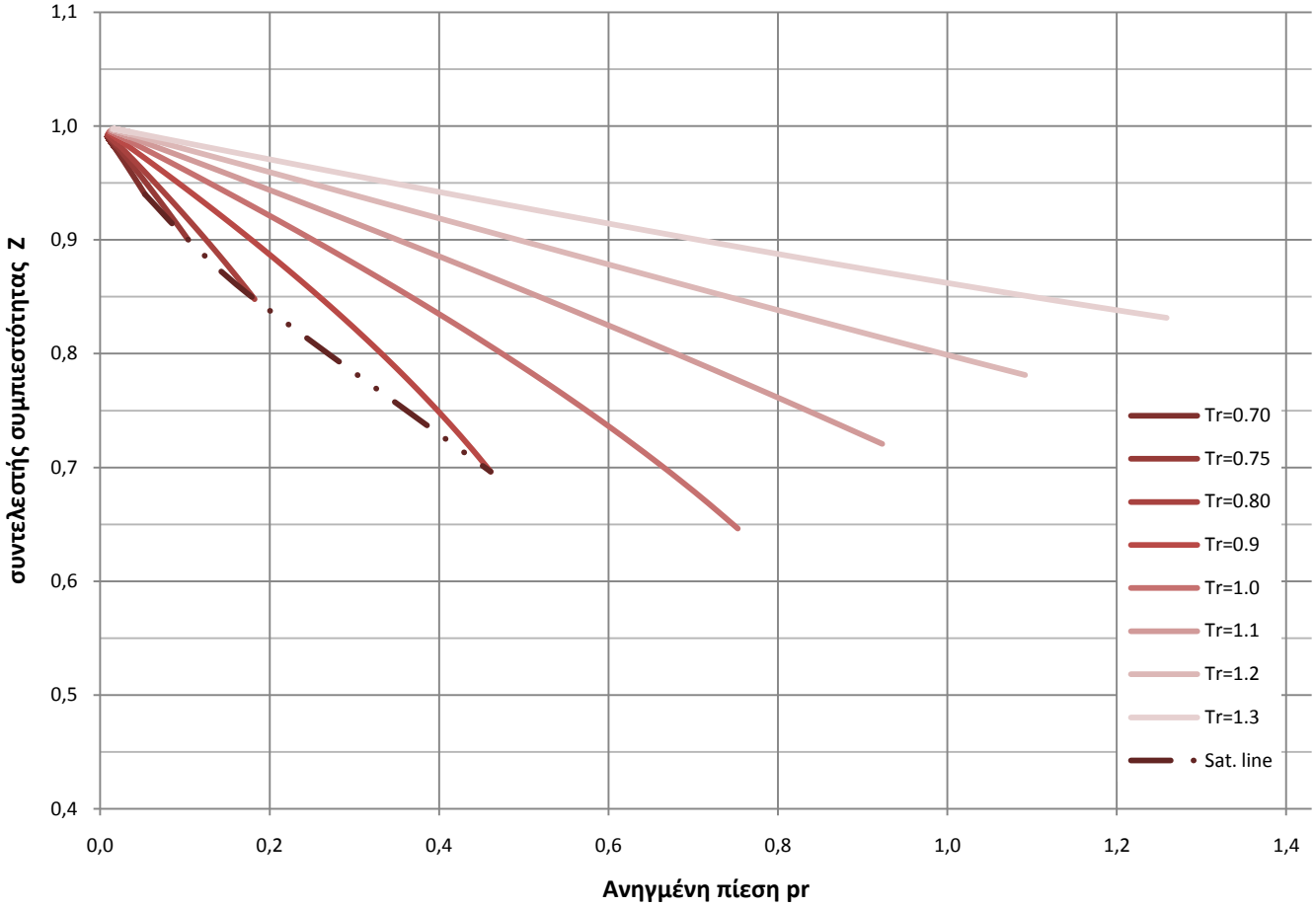
Σχήμα Α.7: *Διάγραμμα λόγου γ συναρτήσει της θερμοκρασίας και της πίεσης για το R407b*



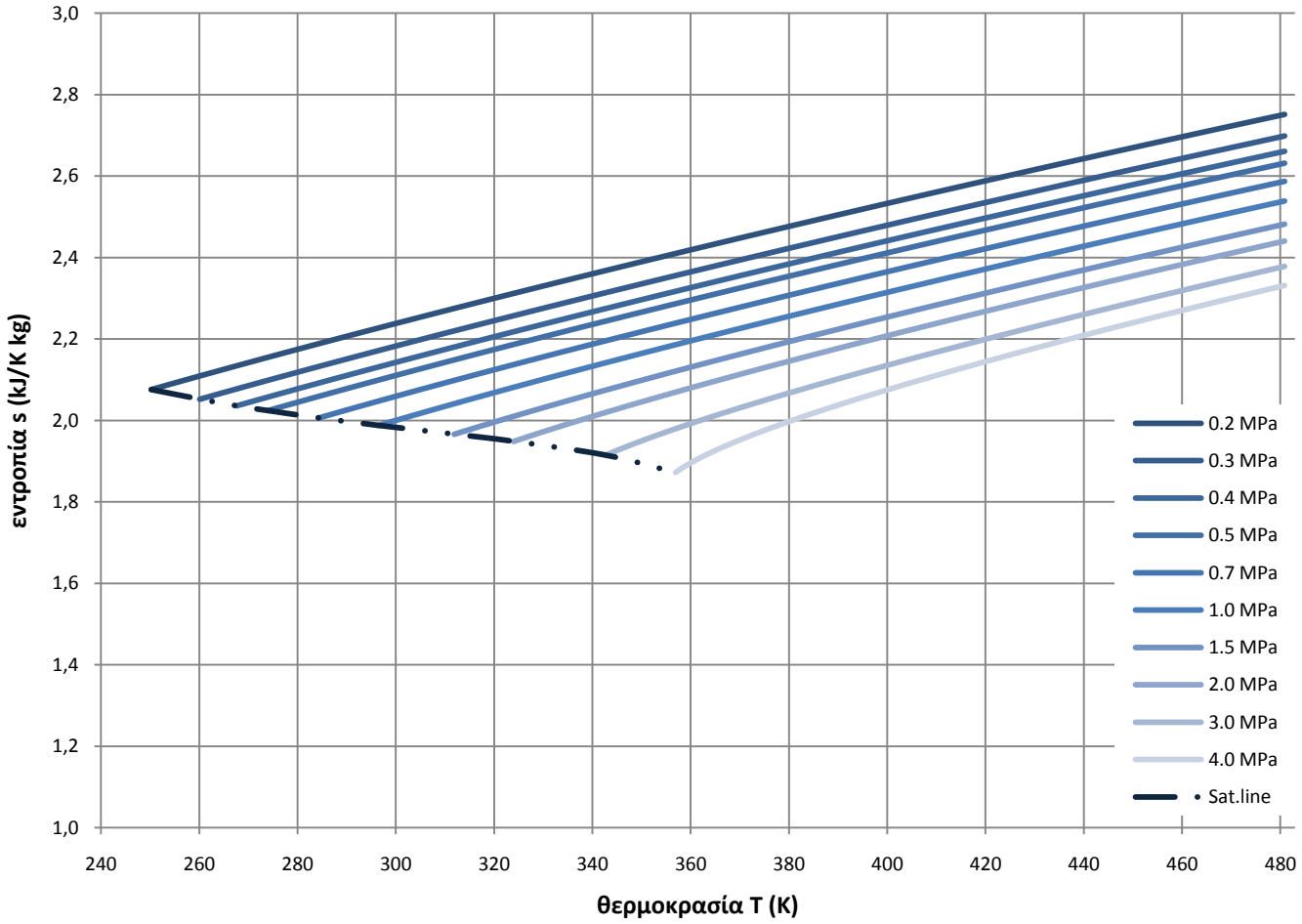
Σχήμα Α.8: *Διάγραμμα ταχύτητας ήχου α συναρτήσει της θερμοκρασίας και της πίεσης για το R407b*



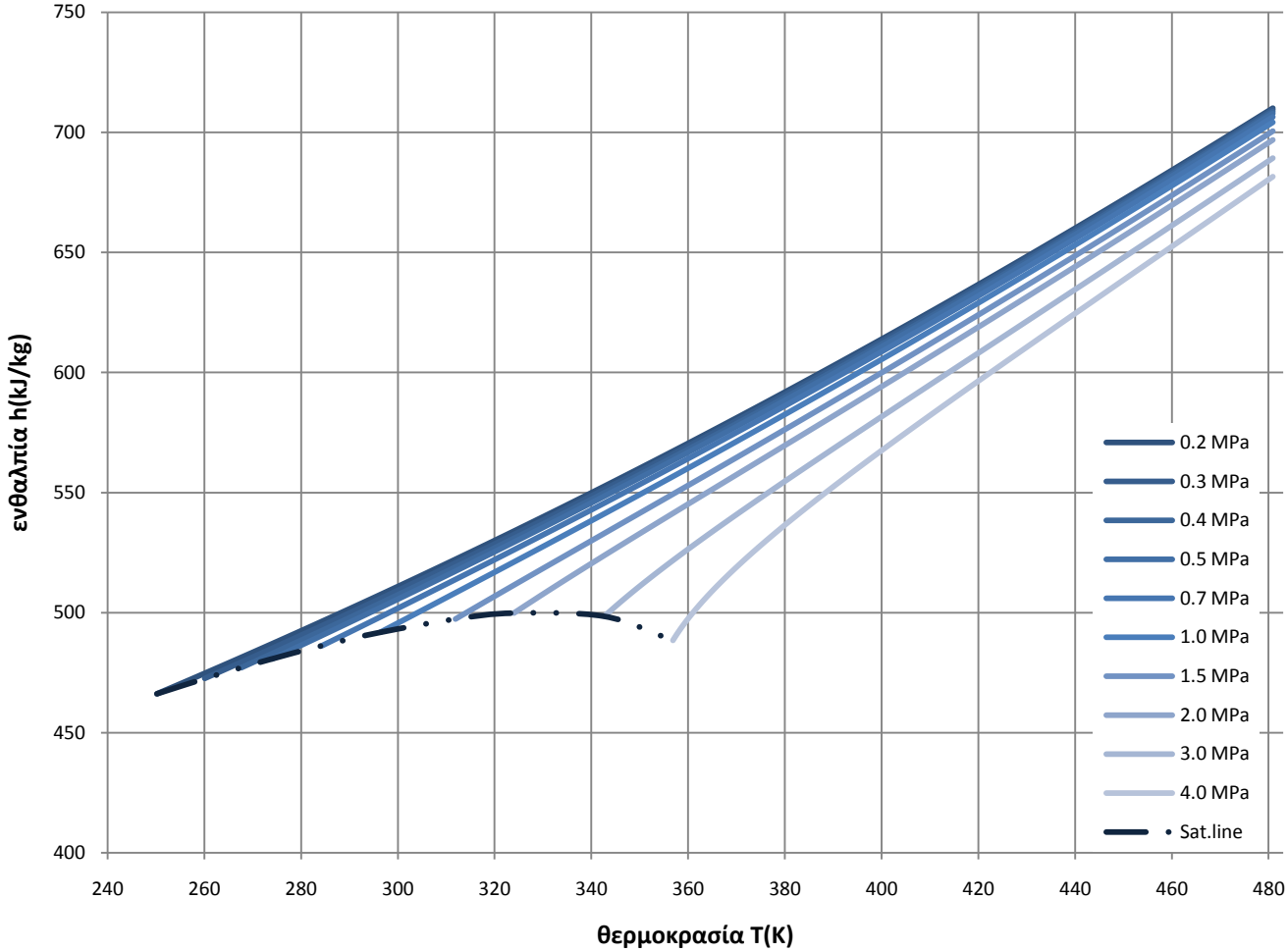
Σχήμα Α.9: *Διάγραμμα συντελεστή συμπιεστότητας Z συναρτήσει της ανηγμένης θερμοκρασίας και της ανηγμένης πίεσης για το R407b*



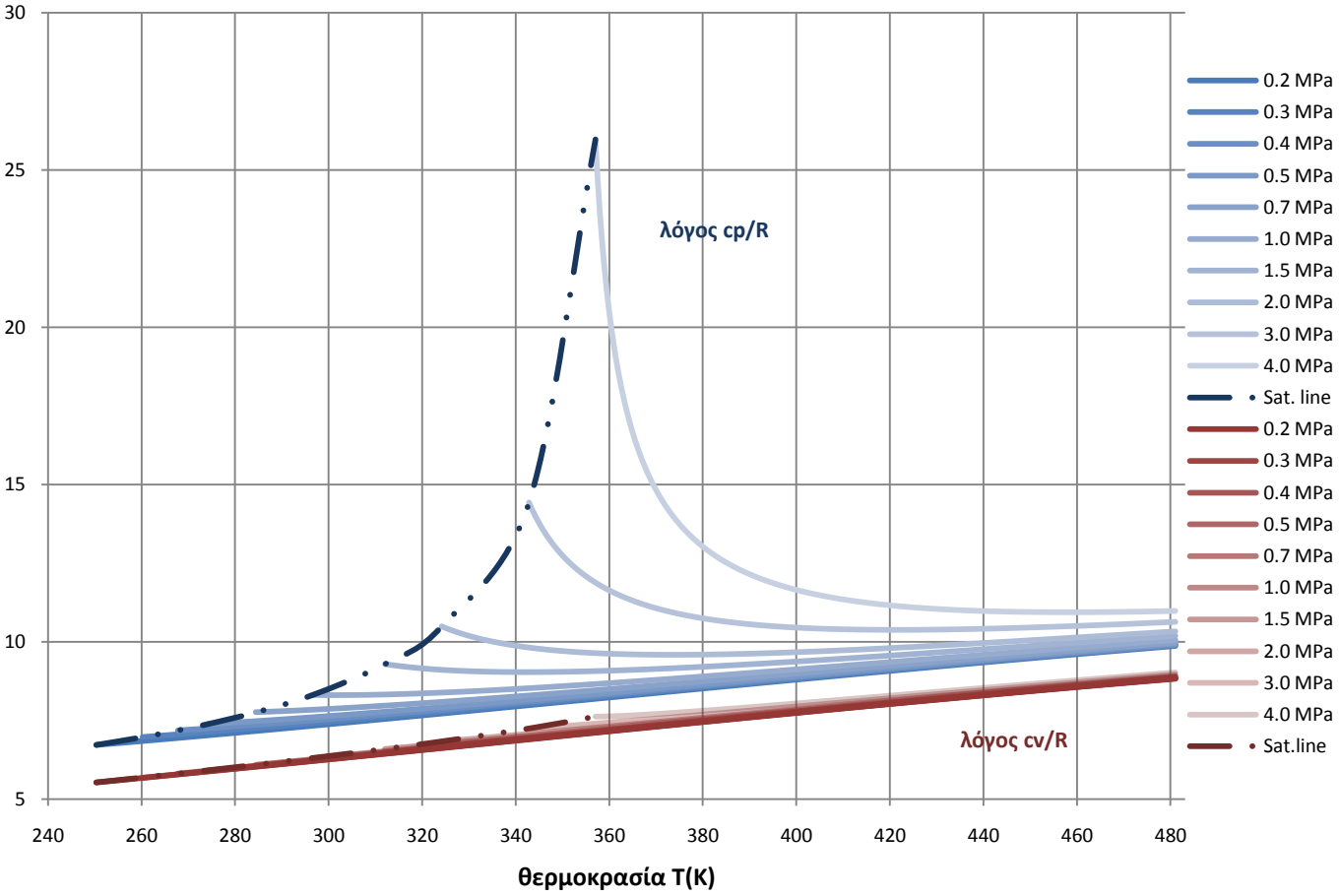
Σχήμα Α.10: *Διάγραμμα εντροπίας συναρτήσει της θερμοκρασίας και της πίεσης για το R152a/R125/R32*



Σχήμα Α.11: *Διάγραμμα ενθαλπίας συναρτήσει της θερμοκρασίας και της πίεσης για το R152a/R125/R32*

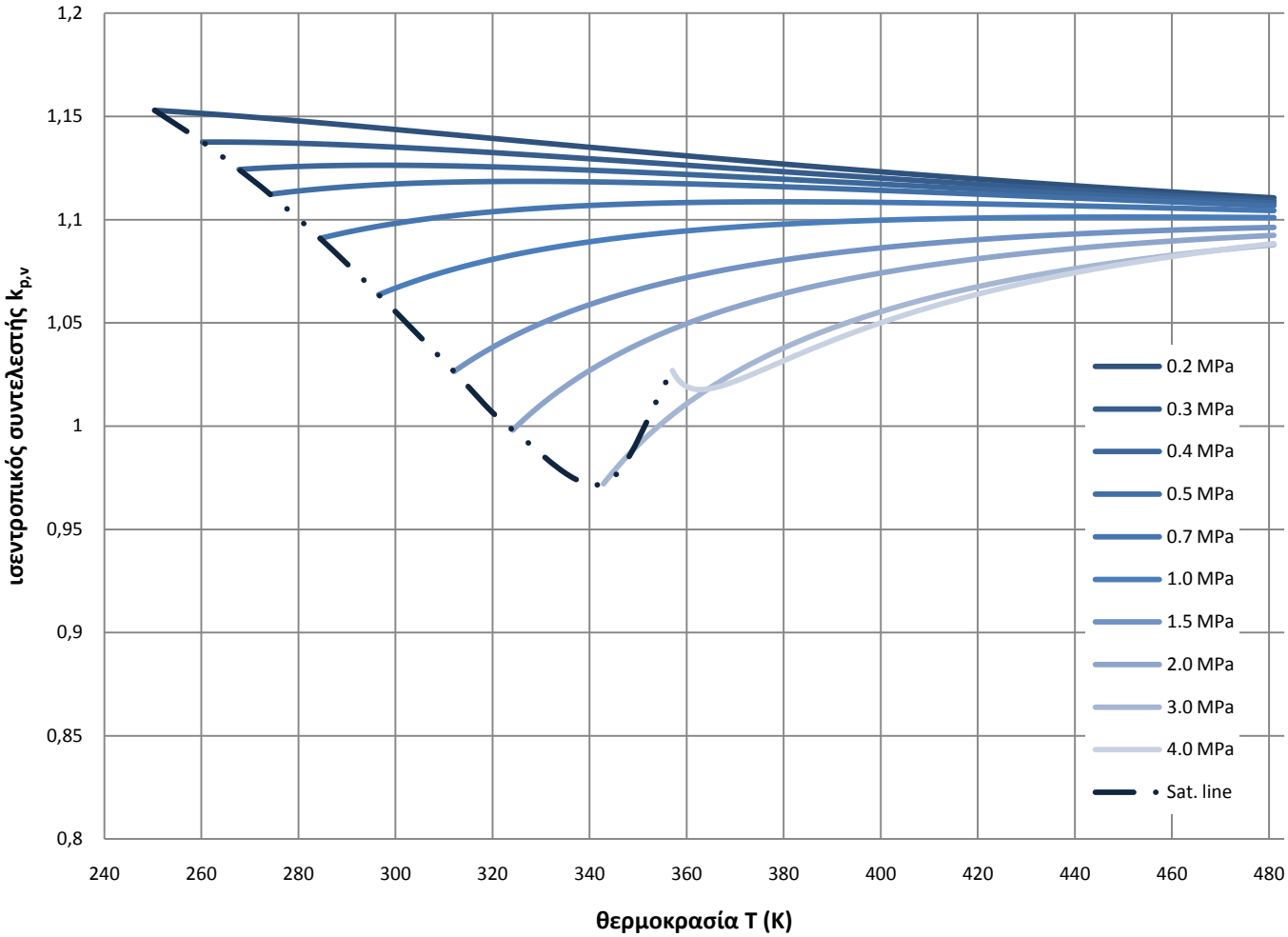


Σχήμα Α.12: *Διαγράμματα λόγων c_p / R και c_v / R ς συναρτήσει της θερμοκρασίας και της πίεσης για το R152a/R125/R32*



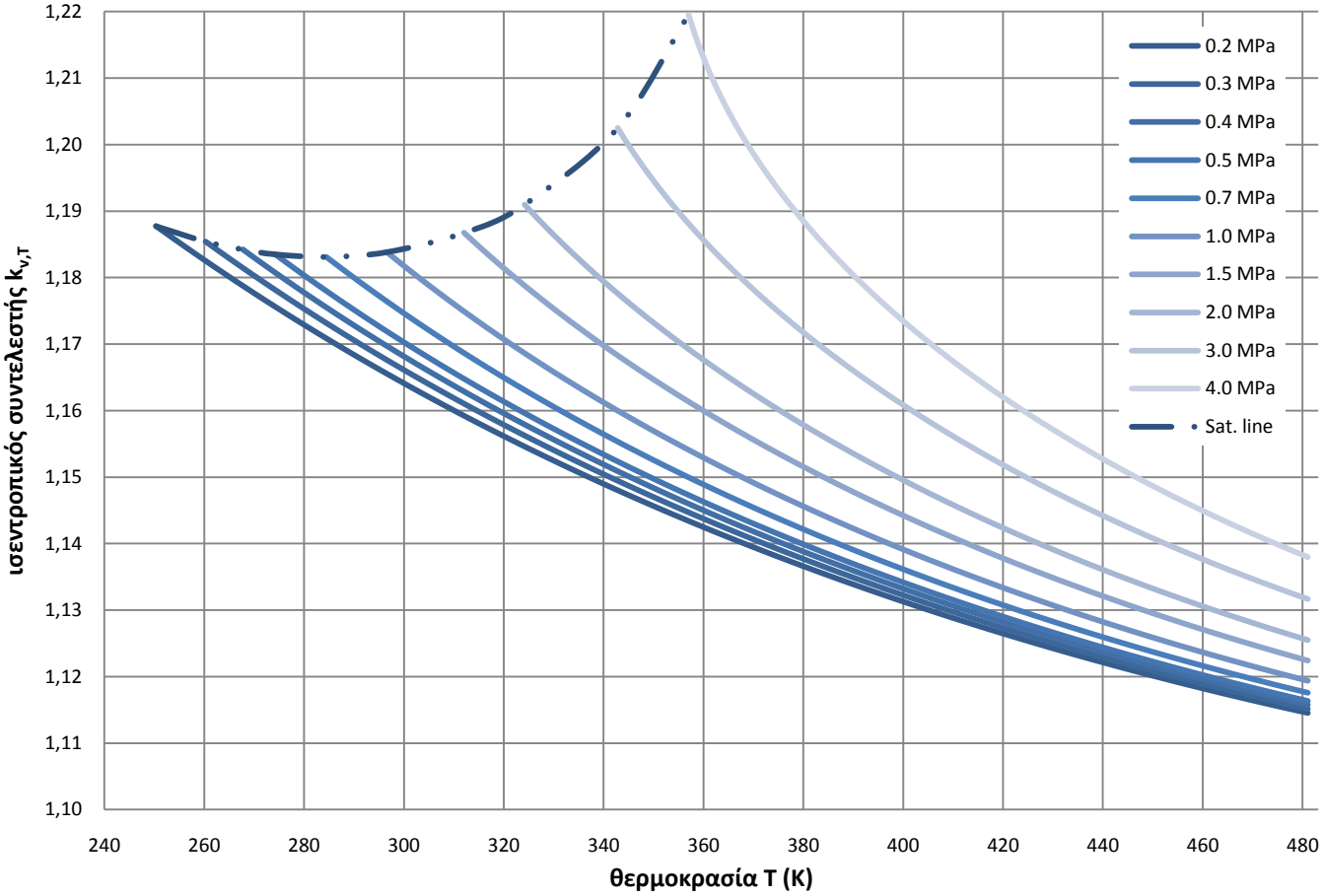
Τα ιδανικά αέρια εμφανίζουν διαφορά $\frac{c_p}{R}-\frac{c_v}{R}=1$, σχέση που επαληθεύεται για το αέριο R407b σε πιέσεις μικρότερες των 0,2MPa, όπως φαίνεται στο σχήμα Α.12.

Σχήμα Α.13: *Διάγραμμα ισεντροπικού συντελεστή $k_{p,v}$ συναρτήσει της θερμοκρασίας και της πίεσης για το R152a/R125/R32*

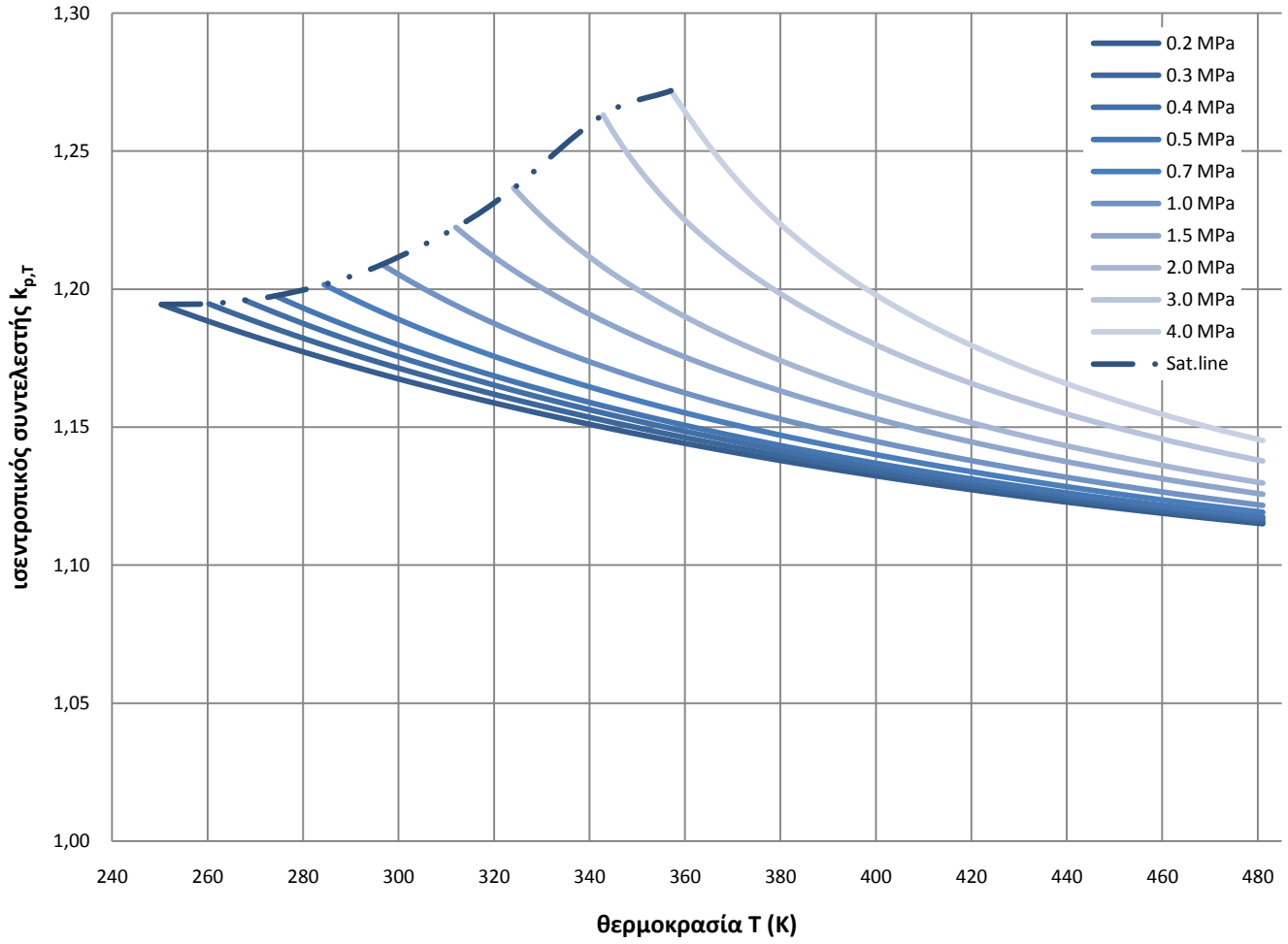


Η ανώμαλη συμπεριφορά του συντελεστή για 4MPa και θερμοκρασίες 355K - 365K, όπου ο όγκος παίρνει τιμές 0,0057 m^3/kg - 0,0067 m^3/kg, ερμηνεύεται από τα μεγάλα τοπικά σφάλματα άνω του 8%. Οι τοπικές αυτές αποκλίσεις οφείλονται εκτός από την γειτνίαση με τη διφασική περιοχή (παράγοντας σφάλματος άλλωστε για κάθε ισόθλιπτη καμπύλη κοντά στη γραμμή κορεσμού) στην εγγύτητα του κρίσιμου σημείου (T_c=367,93K, v_c=0,0025m^3/kg). Για μικρότερες τιμές πίεσης ο όγκος κορεσμού είναι μεγαλύτερος και αισθητά απομακρυσμένος από τον κρίσιμο, με αποτέλεσμα μικρότερα σχετικά σφάλματα. Ενδεικτικά για 3,5MPa ο όγκος κορεσμού είναι v_{sat}=0,0072m^3/kg και ο συντελεστής $k_{p,v}$ παρουσιάζει σφάλμα 3% ενώ για 3MPa ο όγκος κορεσμού είναι v_{sat}=0,0089m^3/kg και το σφάλμα του ίδιου συντελεστή έχει πέσει στο 1,3%.

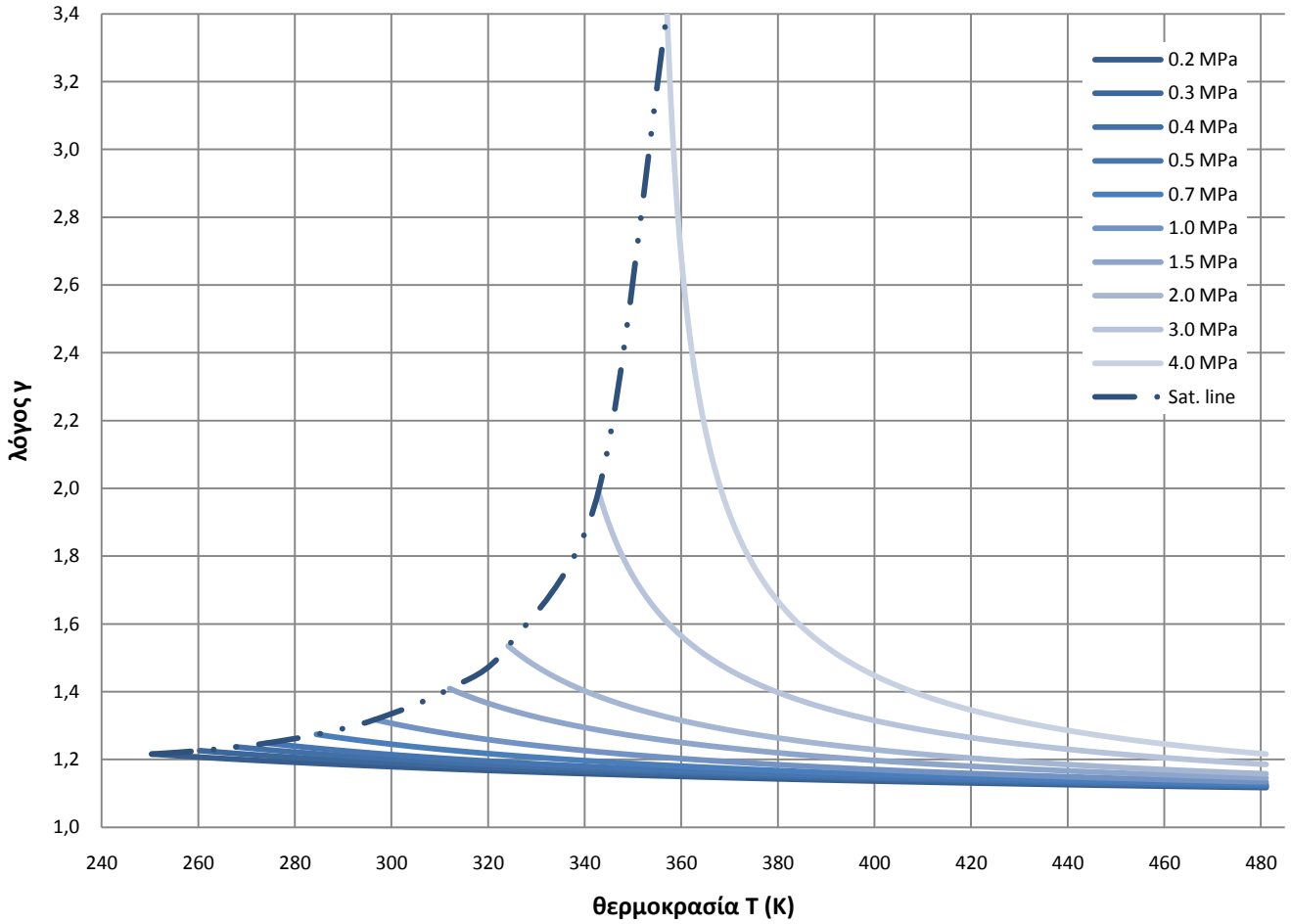
Σχήμα Α.14: *Διάγραμμα ισεντροπικού συντελεστή $k_{v,T}$ συναρτήσει της θερμοκρασίας και της πίεσης για το R152a/R125/R32*



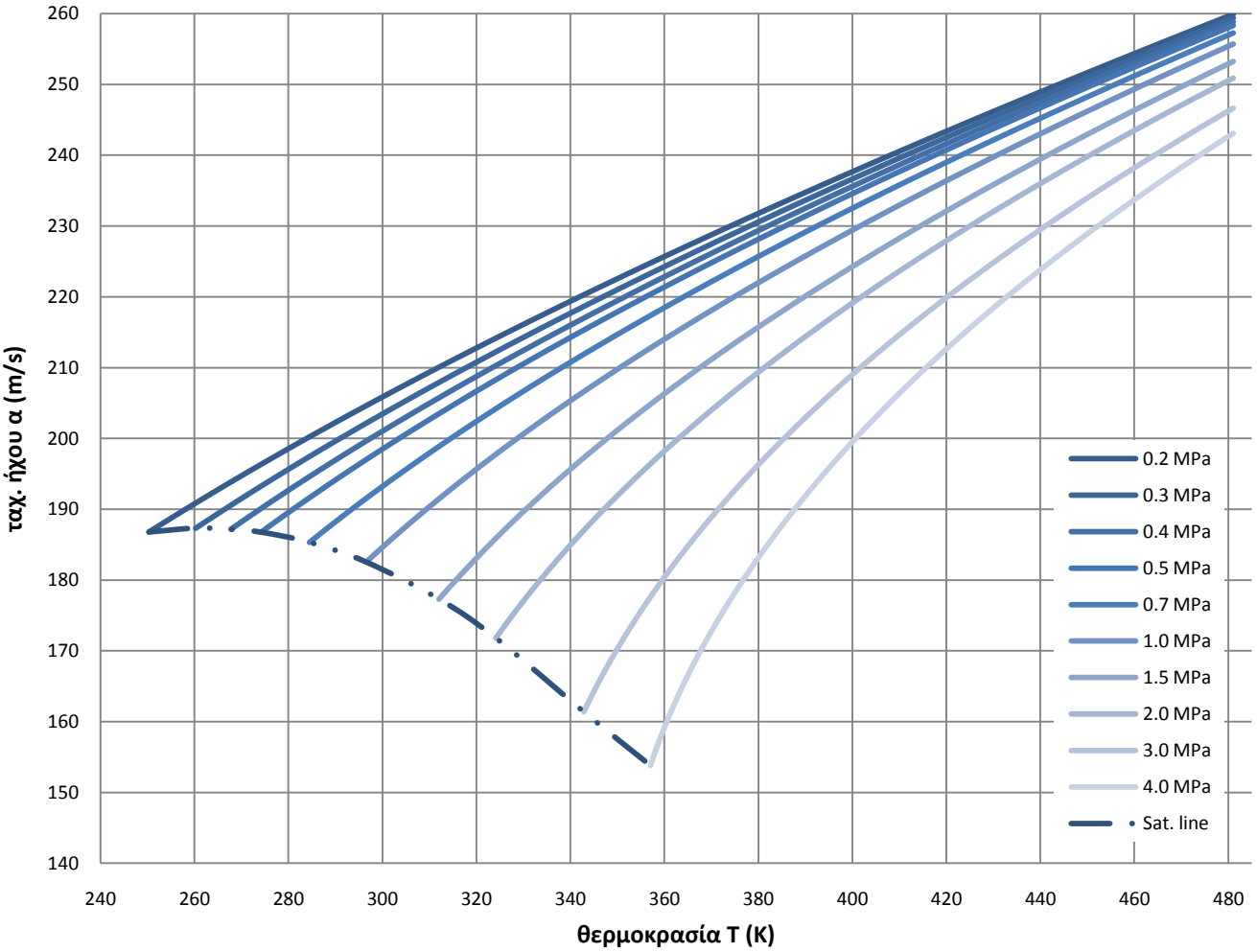
Σχήμα Α.15: *Διάγραμμα ισεντροπικού συντελεστή $k_{p,T}$ συναρτήσει της θερμοκρασίας και της πίεσης για το R152a/R125/R32*



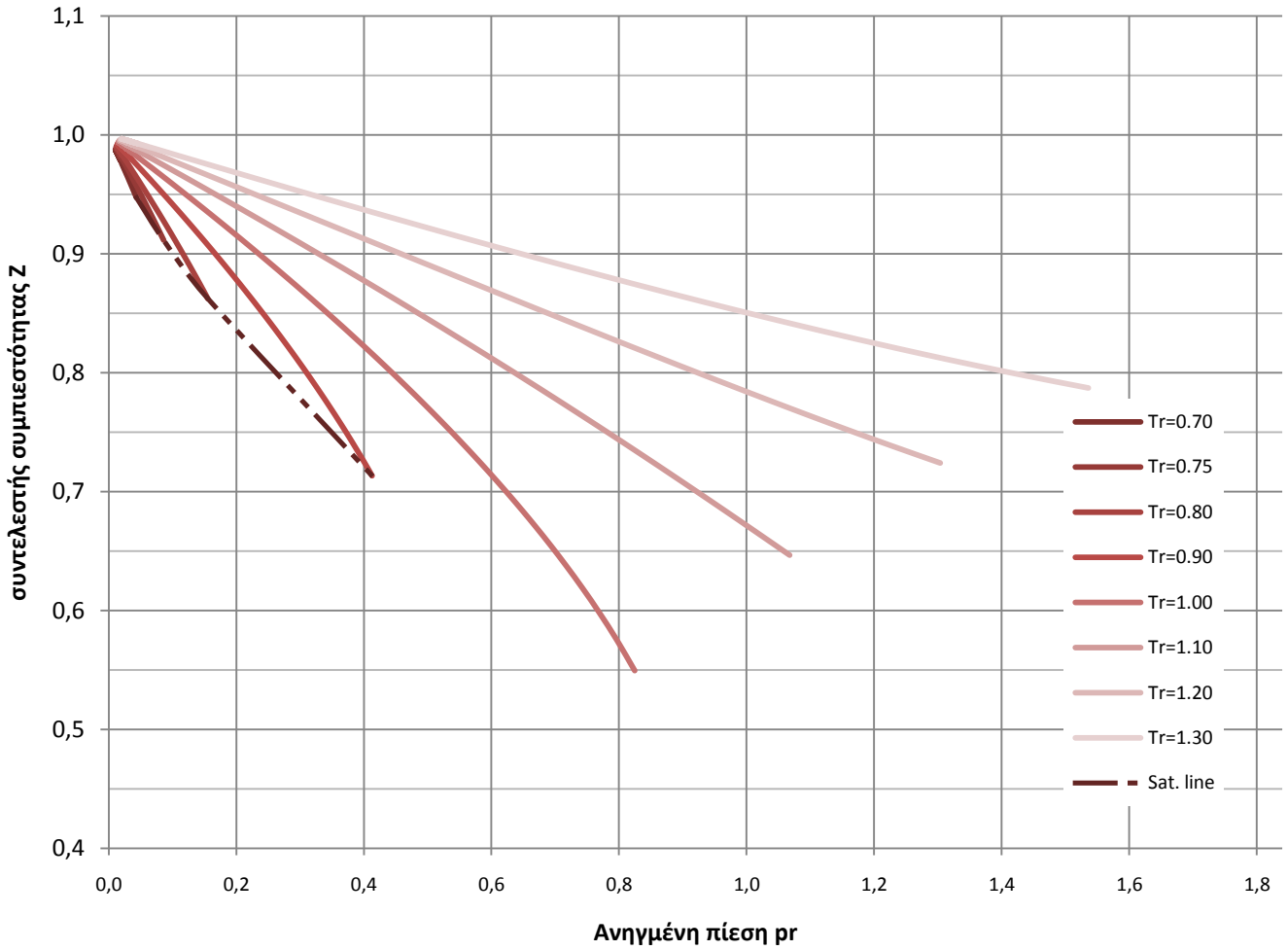
Σχήμα Α.16: *Διάγραμμα λόγου γ συναρτήσει της θερμοκρασίας και της πίεσης για το R152a/R125/R32*



Σχήμα Α.17: *Διάγραμμα ταχύτητας ήχου α συναρτήσει της θερμοκρασίας και της πίεσης για το R152a/R125/R32*

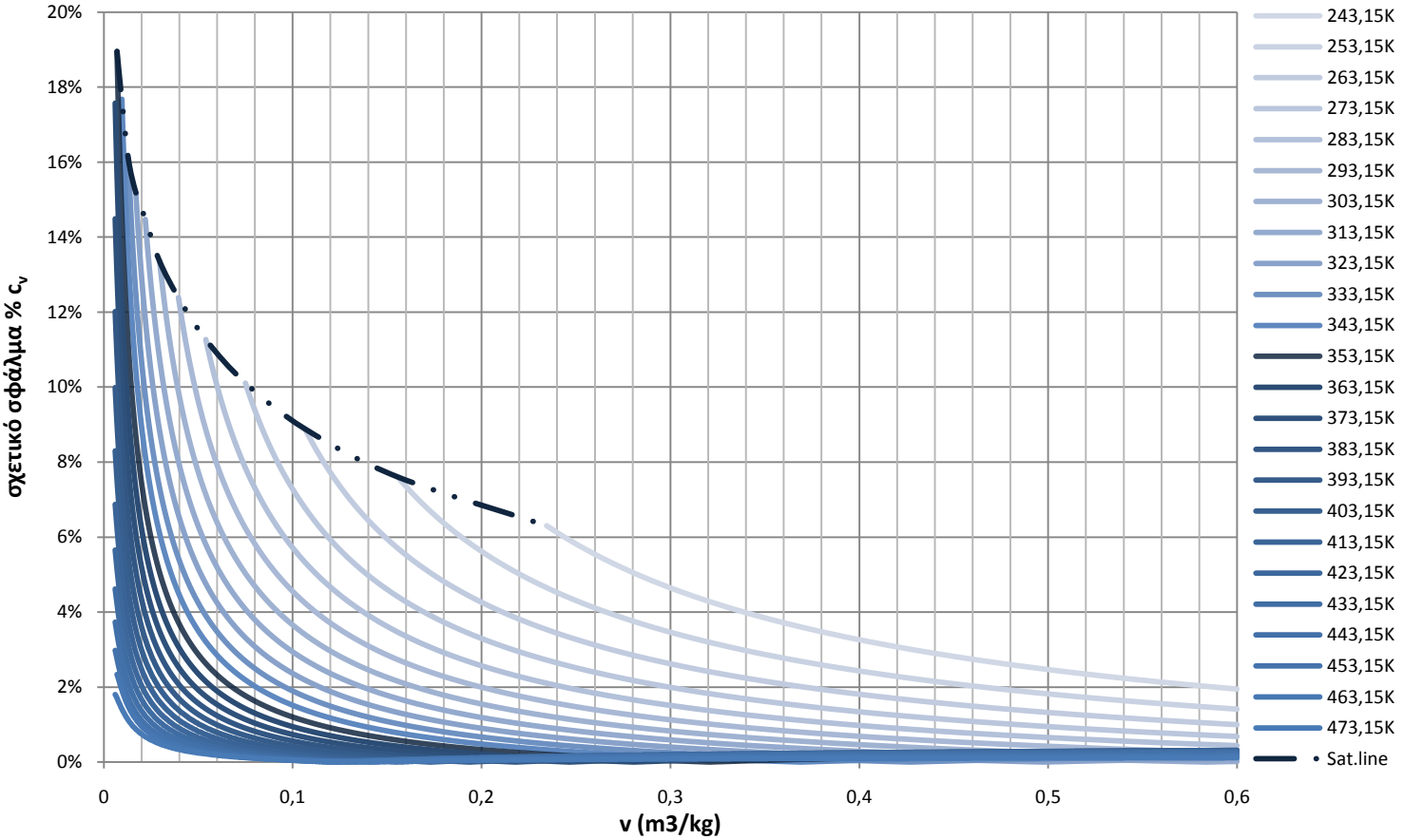


Σχήμα Α.18: *Διάγραμμα συντελεστή συμπιεστότητας συναρτήσει της ανηγμένης θερμοκρασίας και της ανηγμένης πίεσης για το R152a/R125/R32*

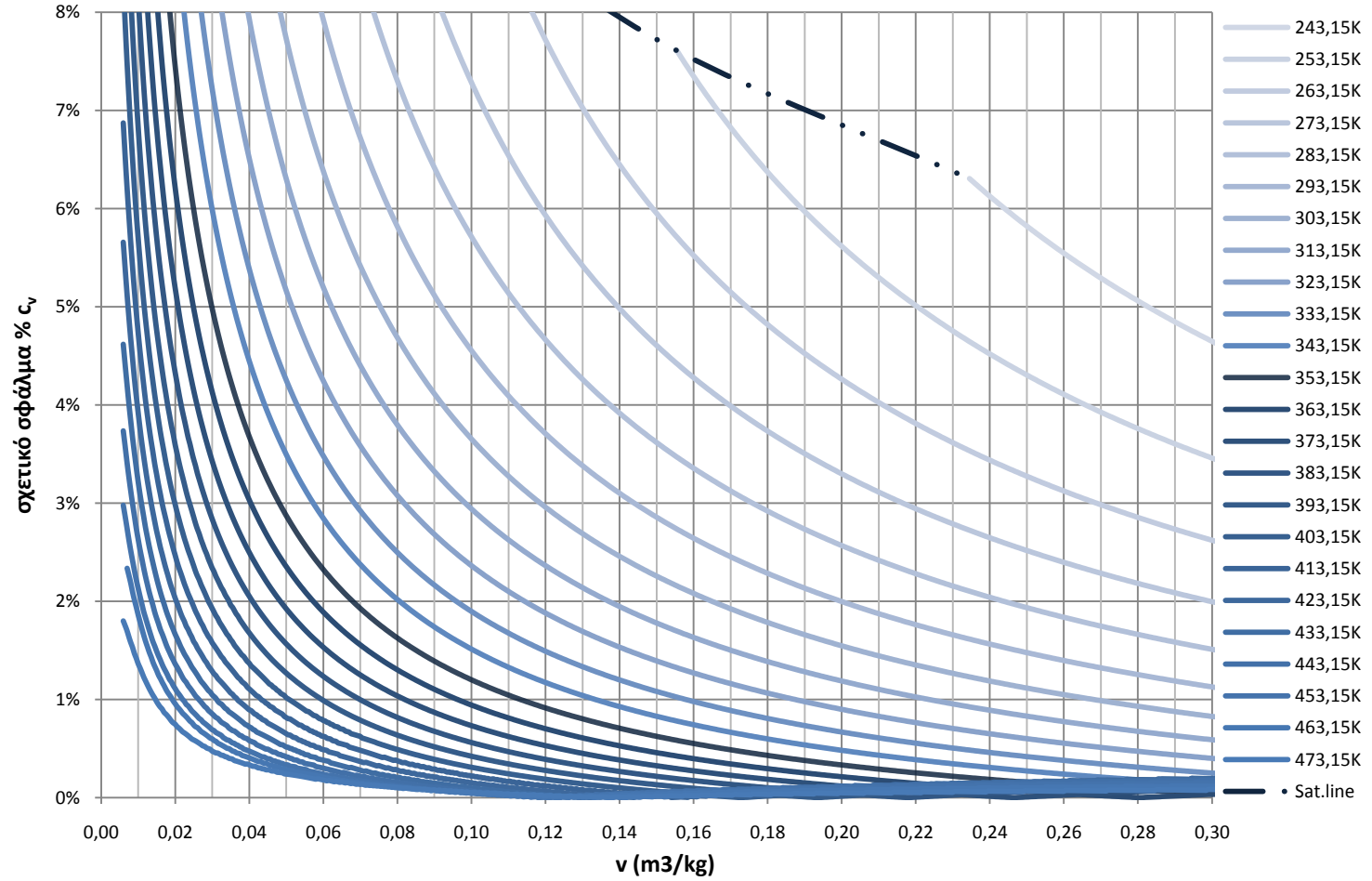


ΠΑΡΑΡΤΗΜΑ Β : Διαγράμματα σφαλμάτων

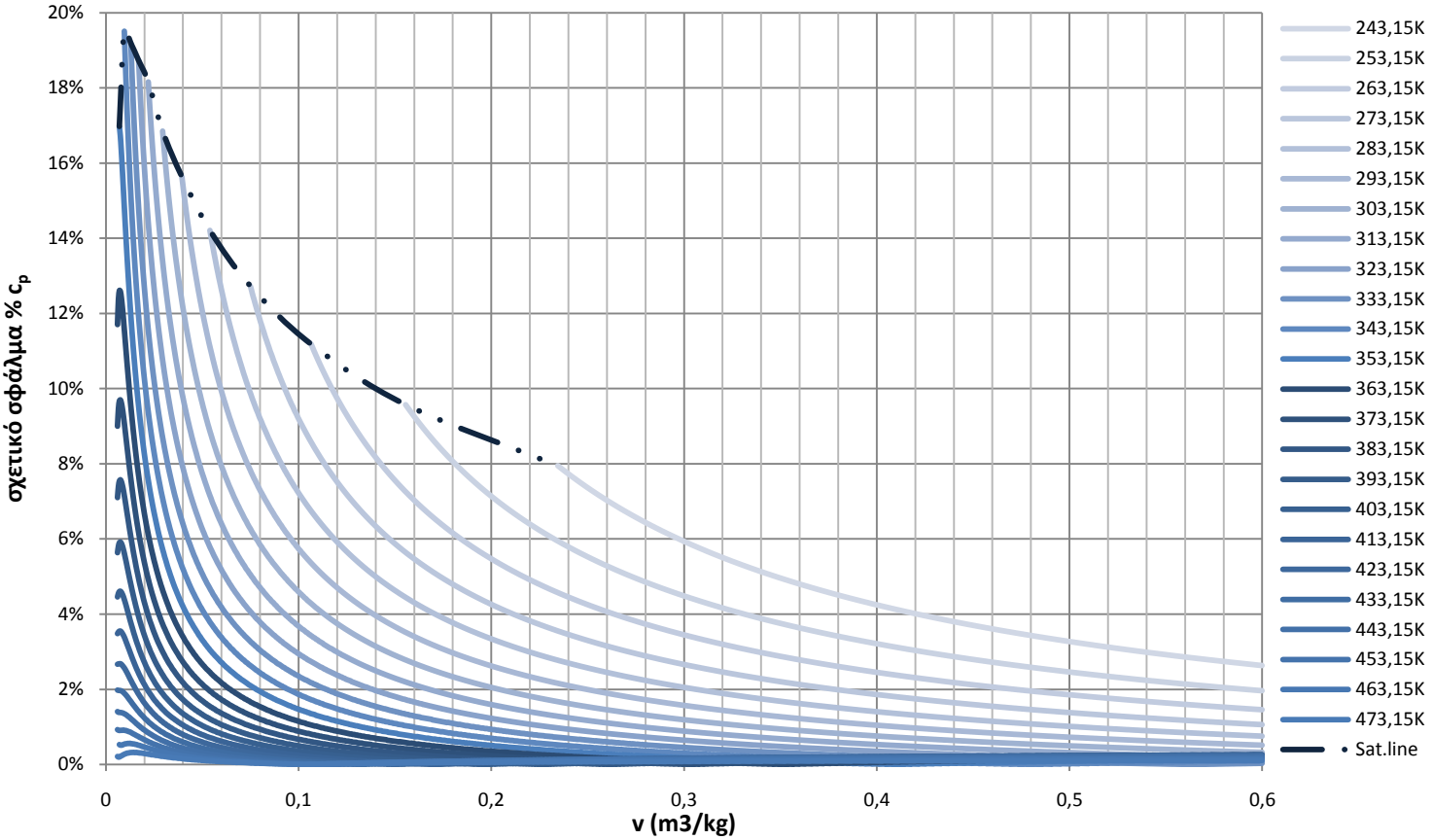
Σχήμα Β.1: *Διάγραμμα σχετικών σφαλμάτων της ειδικής θερμοχωρητικότητας υπό σταθερό όγκο c_v συναρτήσει της θερμοκρασίας και του όγκου για το R152a/R125/R32 για εύρος όγκου 0,006-0,6m^3/kg*



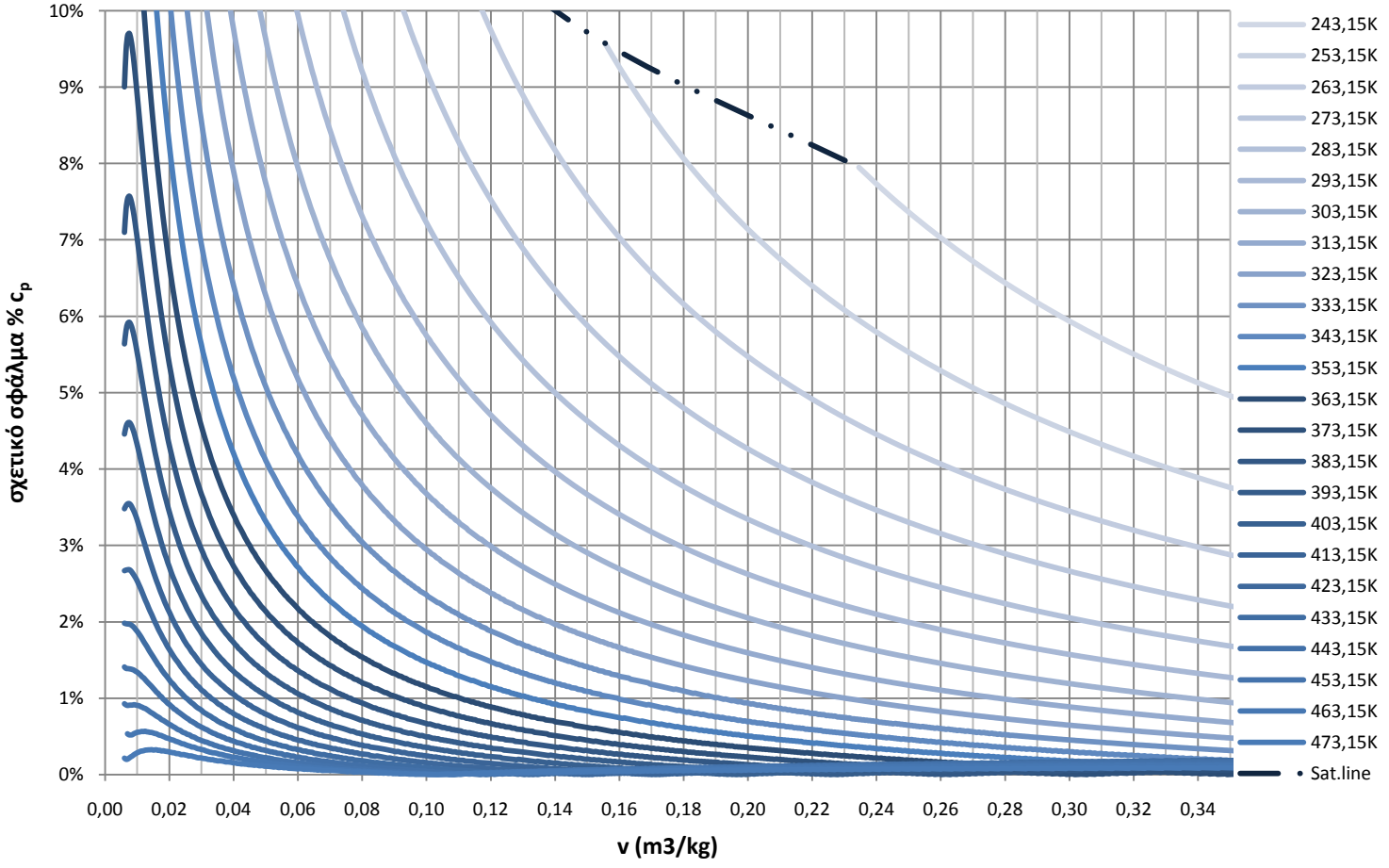
Σχήμα Β.2: *Διάγραμμα σχετικών σφαλμάτων της ειδικής θερμοχωρητικότητας υπό σταθερό όγκο c_v συναρτήσει της θερμοκρασίας και του όγκου για τοR152a/R125/R32 για εύρος όγκου 0,006-0,3m^3/kg*



Σχήμα Β.3: *Διάγραμμα σχετικών σφαλμάτων της ειδικής θερμοχωρητικότητας υπό σταθερή πίεση c_p συναρτήσει της θερμοκρασίας και του όγκου για το R152a/R125/R32 για εύρος όγκου 0,006-0,6m^3/kg*

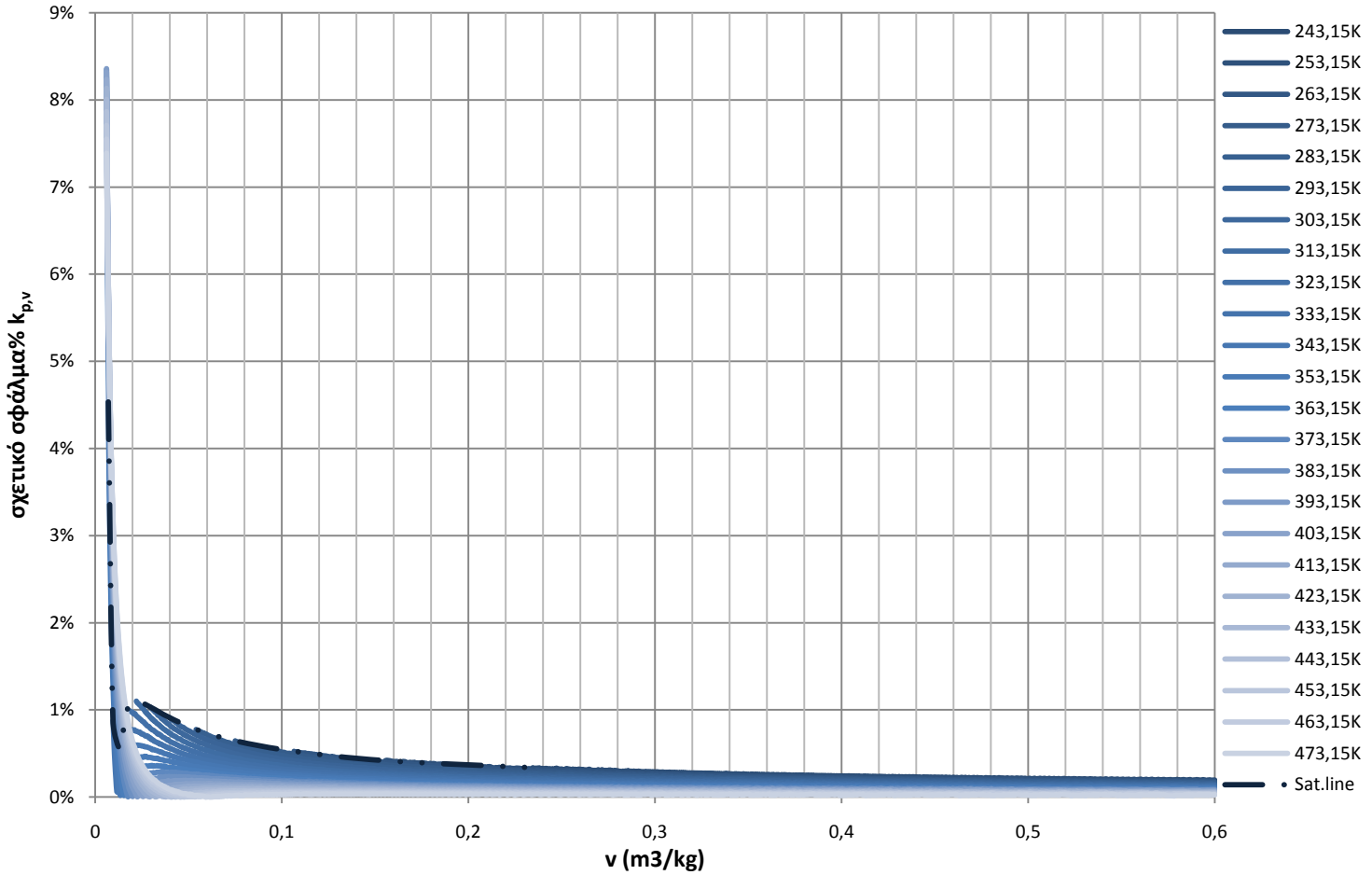


Σχήμα Β.4: *Διάγραμμα σχετικών σφαλμάτων της ειδικής θερμοχωρητικότητας υπό σταθερή πίεση c_p συναρτήσει της θερμοκρασίας και του όγκου για το R152a/R125/R32 για εύρος όγκου 0,006-0,35m^3/kg*

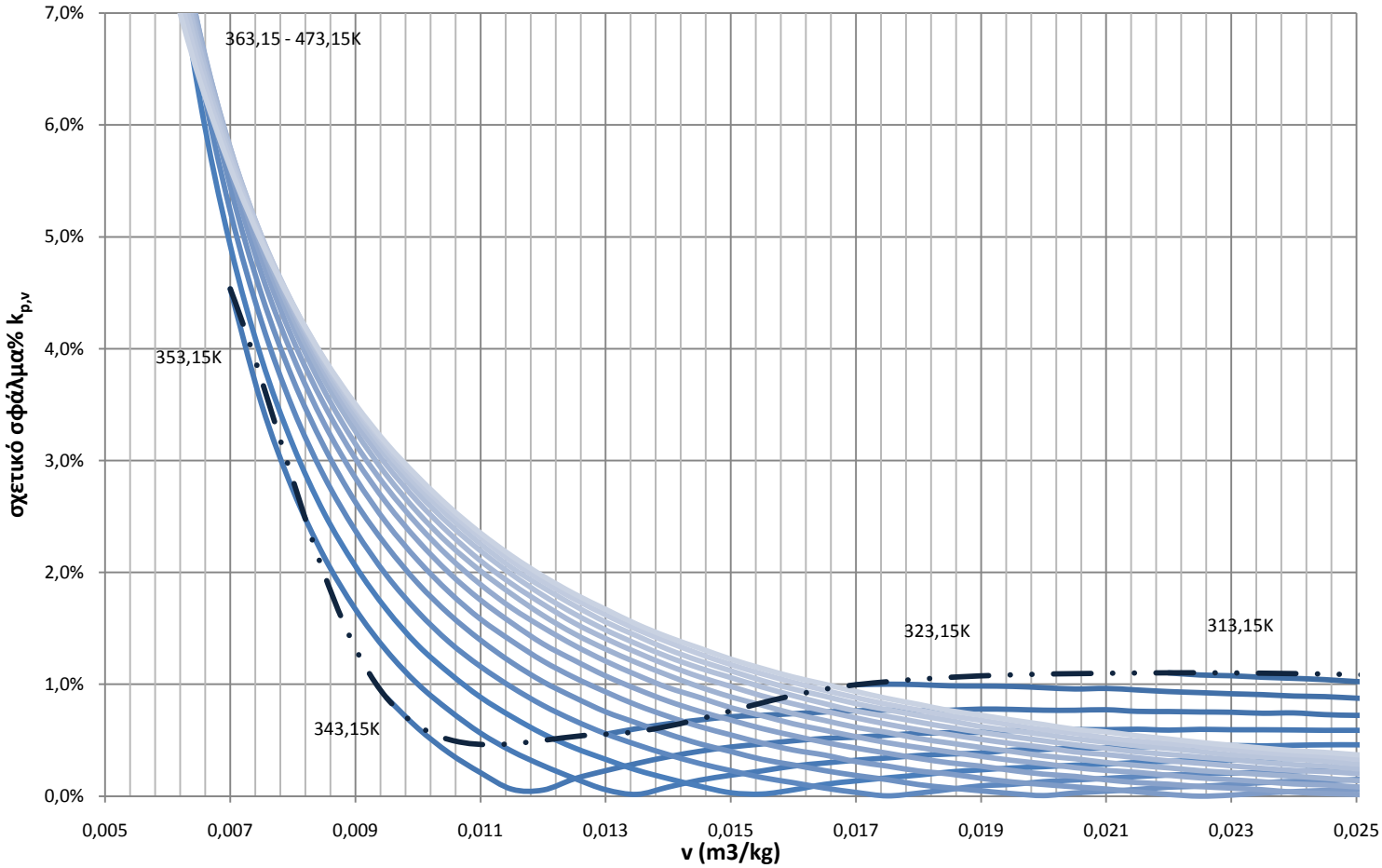


Ενδιαφέρον παρουσιάζει η κορύφωση του σφάλματος για όγκους ελαφρά μεγαλύτερους από τον οριακό 0,006m^3/kg της μελέτης. Αυτό οφείλεται στο μοντέλο προσομοίωσης της συμπεριφοράς του αερίου, δηλαδή στη Peng-Robinson.

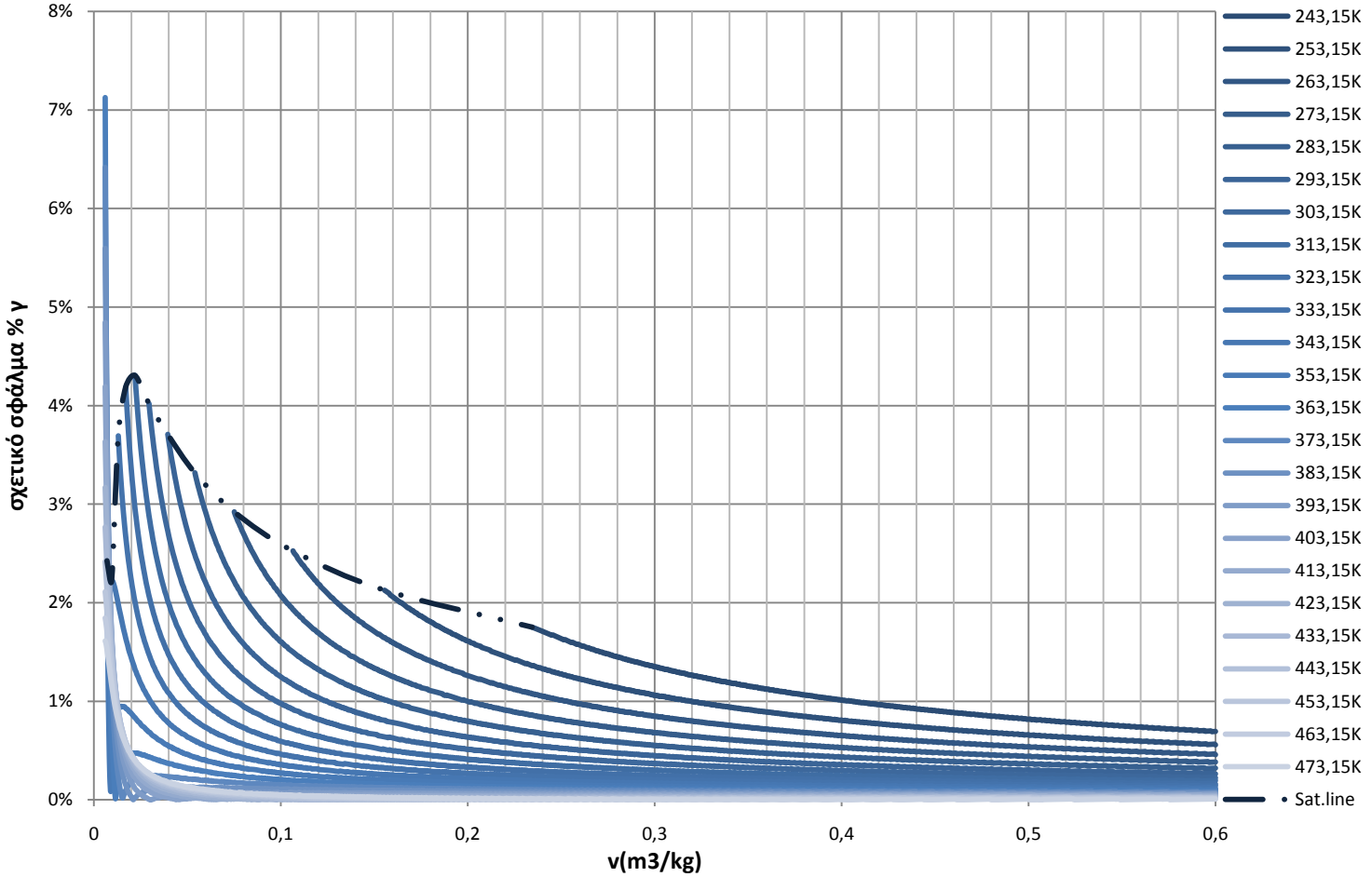
Σχήμα Β.5: *Διάγραμμα σχετικών σφαλμάτων του ισεντροπικού συντελεστή $k_{p,v}$ συναρτήσει της θερμοκρασίας και του όγκου για το R152a/R125/R32 για εύρος όγκου 0,006-0,6m^3/kg*



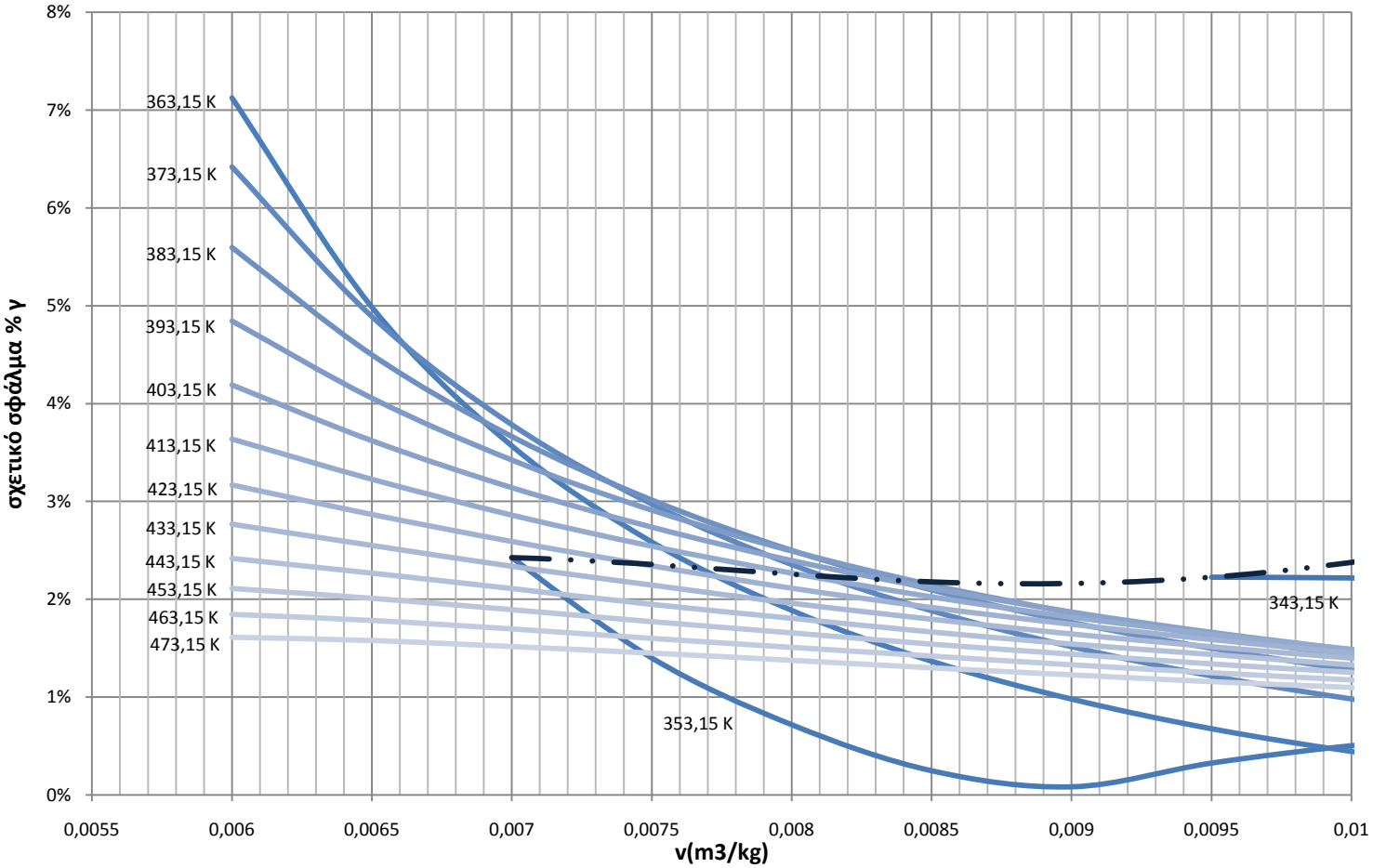
Σχήμα Β.6: *Διάγραμμα σχετικών σφαλμάτων του ισεντροπικού συντελεστή $k_{p,v}$ συναρτήσει της θερμοκρασίας και του όγκου για τοR152a/R125/R32 για εύρος όγκου 0,006-0,025m^3/kg*



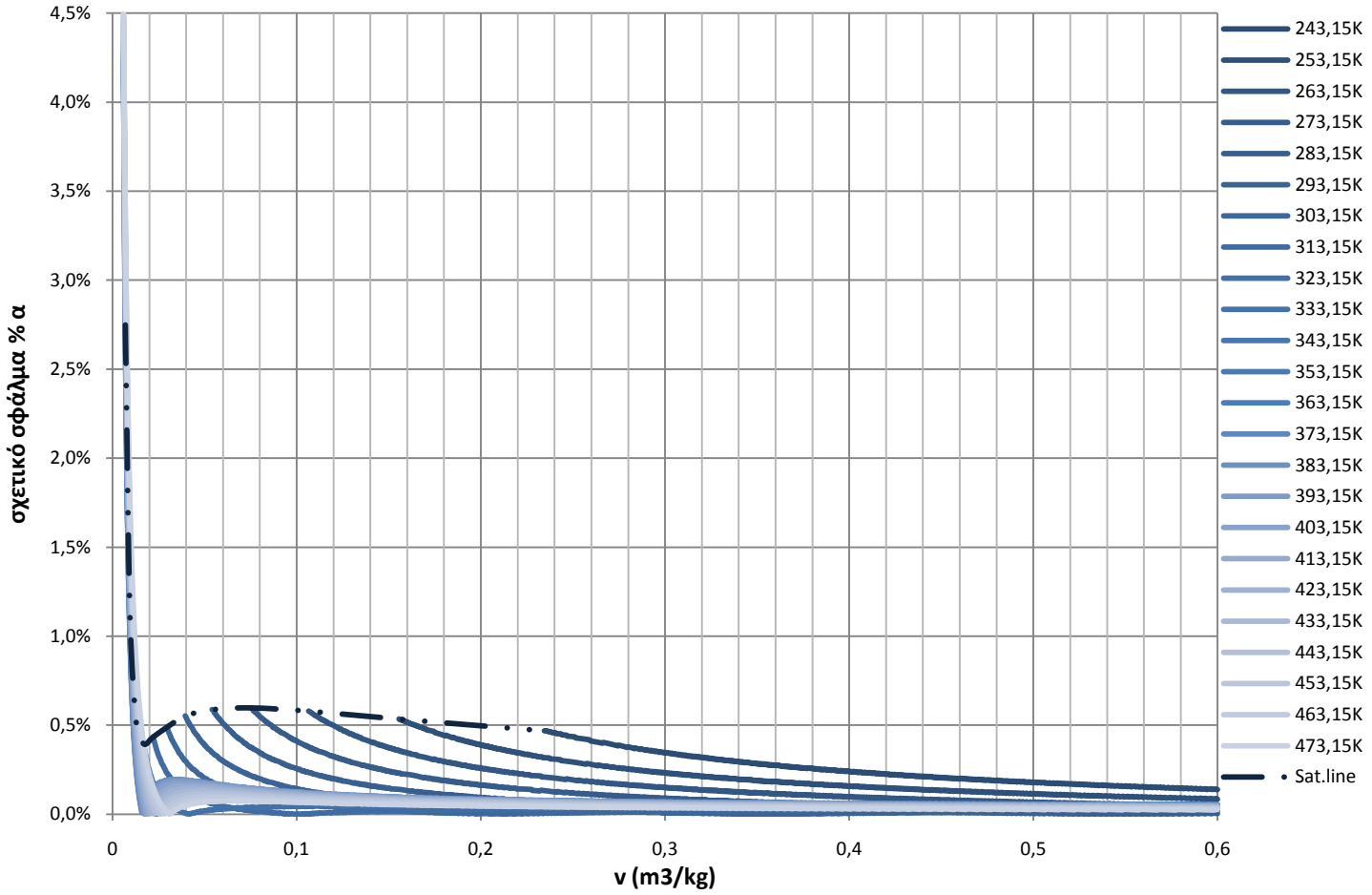
Σχήμα Β.7: *Διάγραμμα σχετικών σφαλμάτων του λόγου γ συναρτήσει της θερμοκρασίας και του όγκου για το R152a/R125/R32 για εύρος όγκου 0,006-0,6m^3/kg*



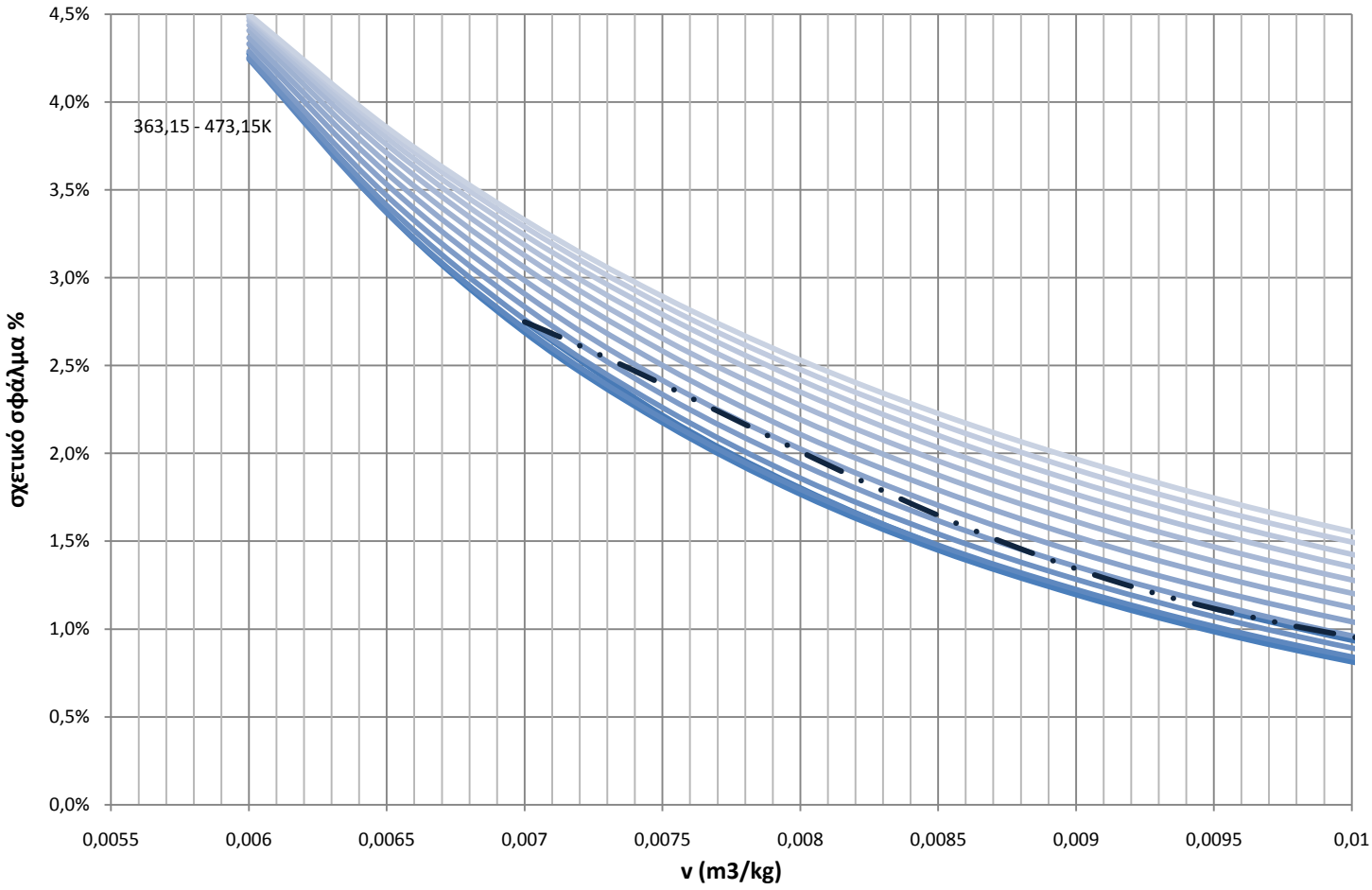
Σχήμα Β.8: *Διάγραμμα σχετικών σφαλμάτων του λόγου γ συναρτήσει της θερμοκρασίας και του όγκου για το R152a/R125/R32 για εύρος όγκου 0,006-0,01m^3/kg*



Σχήμα Β.9: *Διάγραμμα σχετικών σφαλμάτων της ταχύτητας του ήχου α συναρτήσει της θερμοκρασίας και του όγκου για το R152a/R125/R32 για εύρος όγκου 0,006-0,6m^3/kg*



Σχήμα Β.10: *Διάγραμμα σχετικών σφαλμάτων της ταχύτητας του ήχου α συναρτήσει της θερμοκρασίας και του όγκου για το R152a/R125/R32 για εύρος όγκου 0,006-0,01m^3/kg*



ΠΑΡΑΡΤΗΜΑ Γ: Υπολογιστικός κώδικας για το μιγμα R152a/R125/R32

```
// dhmiourgia fakelou refrigeration (‘package’) opou apothikeuetai to arxeio

package refrigeration;

// fortwma to arxeio java (‘class’) gia tis apothikeuseis twn pinakwn sto Excel

import katharo.ExcelFile;

// dhmiourgia arxeio java (‘class’) to opoio tha upoligizei oles tis
// thermophusikes idiothtes

public class PengRobinsonR152aR125R32 {

// dhmiourgia basikhs ‘method’ opou grafontai oles oi kentrikes entoles
// (dhmiourgia pinakwn, apothikeush timwn se aytous apo sunarthseis,
// ektupwseis ktl)

public static void main(String[] args) {

            float temperature = (float) 80;     // bale epithumith timh
                                                // thermokrasias ana 0.1 C
            float volume = (float) 0.007;       // bale epithumith timh
                                                // ogkou ana 0.0005 m3/kg

// apothikeush se pinakes twn statherwn twn ousiwn
            double[] Tc = new double[3];        // krisimes thermokrasies
            Tc[0] = 386.44;
            Tc[1] = 339.45;
            Tc[2] = 351.55;

            double[] pc = new double[3];        // krisimes pieseis
            pc[0] = 4520.0;
            pc[1] = 3630.6;
            pc[2] = 5830.0;

            double[] Vc = new double[3];        // krisimoi ogkoi
            Vc[0] = 179.4837;
            Vc[1] = 211.4;
            Vc[2] = 123.8096;

            double[] w = new double[3];         // krisimoi acentric factors
            w[0] = 0.252851;
            w[1] = 0.294334;
            w[2] = 0.267817;

            double[] ac = new double[3];        //stahera ac ths Peng-Robinson
            ac[0] = 0.457235 * Math.pow(Rmol, 2.0) * Math.pow(Tc[0], 2.0)
                        / (0.000001 * pc[0]);
            ac[1] = 0.457235 * Math.pow(Rmol, 2.0) * Math.pow(Tc[1], 2.0)
                        / (0.000001 * pc[1]);
            ac[2] = 0.457235 * Math.pow(Rmol, 2.0) * Math.pow(Tc[2], 2.0)
                        / (0.000001 * pc[2]);
```

```
            double[] a1 = new double[3];        //stahera a1 ths Peng-Robinson
            a1[0] = 0.826430;
            a1[1] = 0.824698;
            a1[2] = 0.863418;

            double[] a2 = new double[3];        //stahera a2 ths Peng-Robinson
            a2[0] = 0.0;
            a2[1] = 0.0;
            a2[2] = 0.0;

            double[] a3 = new double[3];        //stahera a3 ths Peng-Robinson
            a3[0] = 0.0;
            a3[1] = 0.0;
            a3[2] = 0.0;

            double Rmol = 8.31447215;           // pagosmia stathera aeriwn
                                                // se KJ/KmolK

            double[] b = new double[3];         //stahera b ths Peng-Robinson
            b[0] = 0.077796 * Rmol * Tc[0] / (0.001 * pc[0]);
            b[1] = 0.077796 * Rmol * Tc[1] / (0.001 * pc[1]);
            b[2] = 0.077796 * Rmol * Tc[2] / (0.001 * pc[2]);

            double[] M = new double[3];         // moriaka barh
            M[0] = 66.050;
            M[1] = 120.0215;
            M[2] = 52.035;

            double[] n = new double[3];         // sustash kata maza
            n[0] = 0.48;
            n[1] = 0.18;
            n[2] = 0.34;

            double so = 1.675 + 0.456263644;    // times shmeiou anaforas
            double po = 251.325;
            double To = 273.15;
            double Uo = 292.5 + 162.804172;
            double A = 0.482584348;             // statheres cpideal
            double B = 6.2991266 * 0.0001;
            double C = 4.0487984 * Math.pow(10,-6.0);
            double D = -4.24878898 * Math.pow(10,-9.0);

// upologismos sustashs kata mol

            double[] x = new double[3];
            x[0] = n[0] / (n[0] + n[1] * M[0] / M[1] + n[2] * M[0] /M[2]);
            x[1] = n[1] / (n[1] + n[2] * M[1] / M[2] + n[0] * M[1] /M[0]);
            x[2] = n[2] / (n[2] + n[0] * M[2] / M[0] + n[1] * M[2] /M[1]);
// upologismos sustashs MB,acentric factor, statheras aeriou migmatos

            double Mol = x[0] * M[0] + x[1] * M[1] + x[2] * M[2];
            double wm = w[0] * x[0] + w[1] * x[1] + w[2] * x[2];
            double R = Rmol / Mol;              // se KJ/kgK
```

```
// dhmiourgia pinakwn opou tha apothikeuountai oi times twn thermodunamikwn
// megethwn

            double[] T = new double[2410];
            double[] v = new double[1189];      // m3 / kg
            double[] V = new double[v.length];  // cm3 / mol
            double[] psat = new double[2410];
            double[] vsat = new double[2410];
            double[] cpid = new double[2410];
            double[][] Tr = new double[T.length][3];
            double[] Trm = new double[T.length];
            double[] vrm = new double[v.length];
            double[] Vrm = new double[v.length];
            double[][] a = new double[T.length][3];
            double[][][] k = new double[T.length][3][3];
            double[] k123 = new double[T.length];
            double[][][] aIJ = new double[T.length][3][3];
            double[] a123 = new double[T.length];
            double[] am = new double[T.length];
            double[][] l = l();
            double[][] BIJ = BIJ(b, l);
            double bm = bm(BIJ, x);
            double[][] p = new double[T.length][v.length];
            double[][] prm = new double[T.length][v.length];
            double[][] dadt = new double[T.length][3];
            double[][] dkdt = dkdt();
            double dk123dt = 0.0;
            double[] da123dt = new double[T.length];
            double[][][] daIJdt = new double[T.length][3][3];
            double[] damdt = new double[T.length];
            double[][] dpdt = new double[T.length][v.length];
            double[][] dpdv = new double[T.length][v.length];
            double[][] s = new double[T.length][v.length];
            double[][] h = new double[T.length][v.length];
            double[][] d2adt2 = new double[T.length][3];
            double[] d2a123dt2 = new double[T.length];
            double[][][] d2aIJdt2 = new double[T.length][3][3];
            double[] d2amdt2 = new double[T.length];
            double[][] cv = new double[T.length][v.length];
            double[][] cp = new double[T.length][v.length];
            double[][] kpv = new double[T.length][v.length];
            double[][] ktv = new double[T.length][v.length];
            double[][] kpt = new double[T.length][v.length];
            double[][] g = new double[T.length][v.length];
            double[][] Z = new double[T.length][v.length];
            double[][] atax = new double[T.length][v.length];


// upologismos krisimwn megethwn me tis proteinomenes sxeseis gia thn
// katastatikh eksiswsh Peng Robinson

            double Tcm1 = 0.00;
            double Tcm1Old;
            double Am;
```

```
            do {
                  double[] A1 = A1(Tcm1, Tc, w, pc, Rmol);
                  double[][] K = K(Tcm1);
                  double K123 = K123(Tcm1);
                  double[][] AIJ = AIJ(A1, K);
                  double A123 = A123(K123, A1);
                  Am = Am(AIJ, A123, x);
                  Tcm1Old = Tcm1;
                  Tcm1 = Tcm1(Am, Tc, pc, x, Rmol);

            } while (Math.abs(Tcm1Old - Tcm1) > 0.000001);

            double pcm1 = pcm1(Tc, pc, x, Tcm1);

            double Vcm1 = 1950;
            double Vcm1Old;
            do {
                  Vcm1Old = Vcm1;
                  Vcm1 = Vcm1(Tcm1, pcm1, Am, bm, Rmol, Vcm1Old);

            } while (Math.abs(Vcm1Old - Vcm1) > 0.000001);
            double vcm1 = Vcm1 / (Mol * 1000);

// upologismos krisimwn megethwn me tis proteinomenes sxeseis gia thn
// katastatikh eksiswsh Lee Kesler

            double [][] Tcij=Tcij(Tc);
            double [][] Vcij=Vcij(Vc) ;
            double Vcm2=Vcm2 (Vcij, x);
            double vcm2=Vcm2/(Mol*1000);
            double Tcm2=Tcm2 (Tcij, Vcij, Vcm2, x);
            double pcm2=1000*(0.2905-0.085*wm)*Rmol*Tcm2/Vcm2;

// tupwma sthn othonh twn krisimwm timwn

            System.out.println("Critical values with PR epanalhptikh");
            System.out.println("Tcm  = " + Tcm1 + "  sfalma = 0.57% ");
            System.out.println("pcm  = " + pcm1 + "  sfalma = 4.93% ");
            System.out.println("vcm  = " + vcm1 + "  sfalma = 20.66% ");
            System.out.println("Vcm  = " + Vcm1 + "  sfalma = 20.66% ");
            System.out.println();
            System.out.println("Critical values with Lee Kesler");
            System.out.println("Tcm  = " + Tcm2 + "  sfalma = 0.46% ");
            System.out.println("pcm  = " + pcm2 + "  sfalma = 1.56% ");
            System.out.println("vcm  = " + vcm2 + "  sfalma = 2.56% ");
            System.out.println("Vcm  = " + Vcm2 + "  sfalma = 2.56% ");
            System.out.println();
            System.out.println("Mol  = " + Mol);
            System.out.println("R  = " + R);
            System.out.println("wm  = " + wm);
```

```
// apothikeush stous antistoixous pinakes twn timwn twn megethwn pou
// eksartountai mono apo th T kalwmtas thn katallhlh sunarthsh me to
// katallhlo stoixeio eisodou

            for (int i = 0; i < 2410; i++) {

                  T[i] = 240.15 + 0.1 * i;
                  Tr[i] = tr(T[i], Tc)
                  Trm[i] = trm(T[i], Tcm2);
                  a[i] = a(Tr[i], ac, a1, a2, a3);
                  k[i] = k(T[i]);
                  k123[i] = k123(T[i]);
                  aIJ[i] = aIJ(a[i], k[i]);
                  a123[i] = a123(k123[i], a[i]);
                  am[i] = am(aIJ[i], a123[i], x);
                  dadt[i] = dadt(Tr[i], ac, a1, a2, a3, Tc);
                  da123dt[i] = da123dt(k123[i], a[i], dk123dt, dadt[i]);
                  daIJdt[i] = daIJdt(a[i], k[i], dadt[i], dkdt);
                  damdt[i] = damdt(daIJdt[i], da123dt[i], x);
                  d2adt2[i] = d2adt2(Tr[i], ac, a1, a2, a3, Tc);
                  d2a123dt2[i] = d2a123dt2(k123[i], a[i], dk123dt, dadt[i],
                                    d2adt2[i]);
                  d2aIJdt2[i] = d2aIJdt2(a[i], k[i], dadt[i], dkdt,
                                    d2adt2[i]);
                  d2amdt2[i] = d2amdt2(d2aIJdt2[i], d2a123dt2[i], x);
                  cpid[i]=cpideal(T[i],A,B,C,D);
            }

// apothikeush stous antistoixous pinakes twn timwn twn megethwn pou
// eksartountai mono apo to V kalwmtas thn katallhlh sunarthsh me to
// katallhlo stoixeio eisodou

            for (int j = 0; j < 1189; j++) {

                  v[j] = 0.006 + 0.0005 * j;
                  V[j] = V(v[j], Mol);
                  vrm[j] = vrm(v[j], vcm2);
                  Vrm[j] = Vrm(V[j], Vcm2);
            }

// eisagwgh p kai T koresmenou aeriou

            for (int i = 0; i < 573; i++) {
                  psat[i] = psat(T[i]);
                  vsat[i] = vsat1(T[i]);
            }
            for (int i = 573; i < 1166; i++) {
                  psat[i] = psat(T[i]);
                  vsat[i] = vsat2(T[i]);
            }
            for (int i = 1166; i < 2410; i++) {
                  psat[i] = 0.0;
                  vsat[i] = 0.0;
            }
```

```
// apothikeush stous antistoixous pinakes twn timwn twn megethwn pou
// eksartountai apo T kai V kalwmtas thn katallhlh sunarthsh me to
// katallhlo stoixeio eisodou

            for (int i = 0; i < 2410; i++) {
                  for (int j = 0; j < 1189; j++) {

                        p[i][j] = p(am[i], bm, Rmol, T[i], V[j]);
                        prm[i][j] = prm(p[i][j], pcm2);
                        dpdt[i][j] = dpdt(damdt[i], bm, Rmol, V[j]);
                        dpdv[i][j] = dpdv(am[i], bm, Rmol, T[i], V[j]);
                        s[i][j] = s(damdt[i], bm, Rmol, R, Mol, T[i],
                                    To, V[j],so, po, A, B, C, D);
                        h[i][j] = h(damdt[i], am[i], bm, R, Mol, T[i], To,
                                    V[j], Uo, p[i][j], A, B, C, D);
                        cv[i][j] = cv(d2amdt2[i], bm, R, Mol, T[i], V[j],
                                    A, B,C, D);
                        cp[i][j] = cp(cv[i][j], Mol, T[i], dpdt[i][j],
                                    dpdv[i][j]);
                        kpv[i][j] = kpv(V[j], p[i][j], cv[i][j], cp[i][j],
                                    dpdv[i][j]);
                        ktv[i][j] = ktv(V[j], cv[i][j], dpdt[i][j], Mol);
                        kpt[i][j] = kpt(T[i], p[i][j], cv[i][j], cp[i][j],
                                    dpdt[i][j]);
                        g[i][j] = g(cv[i][j], cp[i][j]);
                        Z[i][j] = Z(p[i][j], V[j], T[i], Rmol);
                        atax[i][j] = atax(V[j], cv[i][j], cp[i][j],
                                    dpdv[i][j],Mol);

// mhdenismos timwn entos difasikhs perioxhs

                        if (v[j] < vsat[i]) {

                              p[i][j] = 0;
                              s[i][j] = 0;
                              h[i][j] = 0;
                              cv[i][j] = 0;
                              cp[i][j] = 0;
                              kpv[i][j] = 0;
                              ktv[i][j] = 0;
                              kpt[i][j] = 0;
                              g[i][j] = 0;
                              Z[i][j] = 0;
                              atax[i][j] = 0;
                        }
                  }
            }
            int t;
            int z;

            temperature = 10 * temperature + 330;
            t = Math.round(temperature);
            volume = 2000 * volume - 12;
            z = Math.round(volume);
```

```
// tupwma sthn othonh olwn twn megethwn sthn eisagwmenh timh T kai V

            System.out.println();
            System.out.println("T    = " + T[t]);
            System.out.println("psat(T) = " + psat[t] + "anwt.sfalma 0.04%");
            System.out.println("vsat(T) = " + vsat[t] + "anwt.sfalma 0.02%");
            System.out.println();
            System.out.println("v    = " + v[z]);
            System.out.println("p    = " + p[t][z]);
            System.out.println("s    = " + s[t][z]);
            System.out.println("h    = " + h[t][z]);
            System.out.println("cv   = " + cv[t][z]);
            System.out.println("cp   = " + cp[t][z]);
            System.out.println("kpv  = " + kpv[t][z]);
            System.out.println("ktv  = " + ktv[t][z]);
            System.out.println("kpt  = " + kpt[t][z]);
            System.out.println("g    = " + g[t][z]);
            System.out.println("Z    = " + Z[t][z]);
            System.out.println("atax = " + atax[t][z]);

// dhmiourgia pinakwn gia elegxo apotelesmatwn

            double[] TT = new double[480];
            double[] vv = new double[480];
            double[] pp = new double[480];
            double[] ss = new double[480];
            double[] hh = new double[480];
            double[] cvv = new double[480];
            double[] cpp = new double[480];
            double[] kpvv = new double[480];
            double[] ktvv = new double[480];
            double[] kptt = new double[480];
            double[] gg = new double[480];
            double[] ZZ = new double[480];
            double[] ataxx = new double[480];
            double[] cpidd = new double[480];

// apothikeush timwn elegxou gia thermokrasia t apo grammh koresmou (z) mexri
// kai th teleutaia timh ogkou (0.006)

            t = 30;
            z = 457;

            for (int j = 0; j < 20; j++) {
                  TT[j] = T[t];
                  vv[j] = v[z];
                  pp[j] = p[t][z];
                  ss[j] = s[t][z];
                  hh[j] = h[t][z];
                  cvv[j] = cv[t][z];
                  cpp[j] = cp[t][z];
                  kpvv[j] = kpv[t][z];
                  ktvv[j] = ktv[t][z];
                  kptt[j] = kpt[t][z];
```

```
        gg[j] = g[t][z];
        ZZ[j] = Z[t][z];
        ataxx[j] = atax[t][z];
        cpidd[j]=cpid[t];
        z = z + 38;
}

t = 130;
z = 299;

for (int j = 20; j < 40; j++) {

        TT[j] = T[t];
        vv[j] = v[z];
        pp[j] = p[t][z];
        ss[j] = s[t][z];
        hh[j] = h[t][z];
        cvv[j] = cv[t][z];
        cpp[j] = cp[t][z];
        kpvv[j] = kpv[t][z];
        ktvv[j] = ktv[t][z];
        kptt[j] = kpt[t][z];
        gg[j] = g[t][z];
        ZZ[j] = Z[t][z];
        ataxx[j] = atax[t][z];
        cpidd[j]=cpid[t];
        z = z + 46;
}

t = 230;
z = 201;
for (int j = 40; j < 60; j++) {

        TT[j] = T[t];
        vv[j] = v[z];
        pp[j] = p[t][z];
        ss[j] = s[t][z];
        hh[j] = h[t][z];
        cvv[j] = cv[t][z];
        cpp[j] = cp[t][z];
        kpvv[j] = kpv[t][z];
        ktvv[j] = ktv[t][z];
        kptt[j] = kpt[t][z];
        gg[j] = g[t][z];
        ZZ[j] = Z[t][z];
        ataxx[j] = atax[t][z];
        cpidd[j]=cpid[t];
        z = z + 51;
}
```

```
t = 330;
z = 138;

for (int j = 60; j < 80; j++) {

      TT[j] = T[t];
      vv[j] = v[z];
      pp[j] = p[t][z];
      ss[j] = s[t][z];
      hh[j] = h[t][z];
      cvv[j] = cv[t][z];
      cpp[j] = cp[t][z];
      kpvv[j] = kpv[t][z];
      ktvv[j] = ktv[t][z];
      kptt[j] = kpt[t][z];
      gg[j] = g[t][z];
      ZZ[j] = Z[t][z];
      ataxx[j] = atax[t][z];
      cpidd[j]=cpid[t];
      z = z + 55;
}

t = 430;
z = 96;

for (int j = 80; j < 100; j++) {

      TT[j] = T[t];
      vv[j] = v[z];
      pp[j] = p[t][z];
      ss[j] = s[t][z];
      hh[j] = h[t][z];
      cvv[j] = cv[t][z];
      cpp[j] = cp[t][z];
      kpvv[j] = kpv[t][z];
      ktvv[j] = ktv[t][z];
      kptt[j] = kpt[t][z];
      gg[j] = g[t][z];
      ZZ[j] = Z[t][z];
      ataxx[j] = atax[t][z];
      cpidd[j]=cpid[t];
      z = z + 57;
}

t = 530;
z = 67;

for (int j = 100; j < 120; j++) {
      TT[j] = T[t];
      vv[j] = v[z];
      pp[j] = p[t][z];
      ss[j] = s[t][z];
      hh[j] = h[t][z];
      cvv[j] = cv[t][z];
```

```
      cpp[j] = cp[t][z];
      kpvv[j] = kpv[t][z];
      ktvv[j] = ktv[t][z];
      kptt[j] = kpt[t][z];
      gg[j] = g[t][z];
      ZZ[j] = Z[t][z];
      ataxx[j] = atax[t][z];
      cpidd[j]=cpid[t];
      z = z + 59;
}

t = 630;
z = 47;

for (int j = 120; j < 140; j++) {

      TT[j] = T[t];
      vv[j] = v[z];
      pp[j] = p[t][z];
      ss[j] = s[t][z];
      hh[j] = h[t][z];
      cvv[j] = cv[t][z];
      cpp[j] = cp[t][z];
      kpvv[j] = kpv[t][z];
      ktvv[j] = ktv[t][z];
      kptt[j] = kpt[t][z];
      gg[j] = g[t][z];
      ZZ[j] = Z[t][z];
      ataxx[j] = atax[t][z];
      cpidd[j]=cpid[t];
      z = z + 60;
}

t = 730;
z = 32;

for (int j = 140; j < 160; j++) {

      TT[j] = T[t];
      vv[j] = v[z];
      pp[j] = p[t][z];
      ss[j] = s[t][z];
      hh[j] = h[t][z];
      cvv[j] = cv[t][z];
      cpp[j] = cp[t][z];
      kpvv[j] = kpv[t][z];
      ktvv[j] = ktv[t][z];
      kptt[j] = kpt[t][z];
      gg[j] = g[t][z];
      ZZ[j] = Z[t][z];
      ataxx[j] = atax[t][z];
      cpidd[j]=cpid[t];
      z = z + 60;
}
```

```
t = 830;
z = 22;

for (int j = 160; j < 180; j++) {
      TT[j] = T[t];
      vv[j] = v[z];
      pp[j] = p[t][z];
      ss[j] = s[t][z];
      hh[j] = h[t][z];
      cvv[j] = cv[t][z];
      cpp[j] = cp[t][z];
      kpvv[j] = kpv[t][z];
      ktvv[j] = ktv[t][z];
      kptt[j] = kpt[t][z];
      gg[j] = g[t][z];
      ZZ[j] = Z[t][z];
      ataxx[j] = atax[t][z];
      cpidd[j]=cpid[t];
      z = z + 61;
}

t = 930;
z = 14;

for (int j = 180; j < 200; j++) {

      TT[j] = T[t];
      vv[j] = v[z];
      pp[j] = p[t][z];
      ss[j] = s[t][z];
      hh[j] = h[t][z];
      cvv[j] = cv[t][z];
      cpp[j] = cp[t][z];
      kpvv[j] = kpv[t][z];
      ktvv[j] = ktv[t][z];
      kptt[j] = kpt[t][z];
      gg[j] = g[t][z];
      ZZ[j] = Z[t][z];
      ataxx[j] = atax[t][z];
      cpidd[j]=cpid[t];
      z = z + 61;
}

t = 1030;
z = 7;

for (int j = 200; j < 220; j++) {
      TT[j] = T[t];
      vv[j] = v[z];
      pp[j] = p[t][z];
      ss[j] = s[t][z];
      hh[j] = h[t][z];
      cvv[j] = cv[t][z];
      cpp[j] = cp[t][z];
```

```
      kpvv[j] = kpv[t][z];
      ktvv[j] = ktv[t][z];
      kptt[j] = kpt[t][z];
      gg[j] = g[t][z];
      ZZ[j] = Z[t][z];
      ataxx[j] = atax[t][z];
      cpidd[j]=cpid[t];
      z = z + 62;
}

t = 1130;
z = 2;
for (int j = 220; j < 240; j++) {
      TT[j] = T[t];
      vv[j] = v[z];
      pp[j] = p[t][z];
      ss[j] = s[t][z];
      hh[j] = h[t][z];
      cvv[j] = cv[t][z];
      cpp[j] = cp[t][z];
      kpvv[j] = kpv[t][z];
      ktvv[j] = ktv[t][z];
      kptt[j] = kpt[t][z];
      gg[j] = g[t][z];
      ZZ[j] = Z[t][z];
      ataxx[j] = atax[t][z];
      cpidd[j]=cpid[t];
      z = z + 62;
}

t = 1230;
z = 0;

for (int j = 240; j < 480; j++) {
      if (j == 260) {
            t = 1330;
            z = 0;
      } else if (j == 280) {
            t = 1430;
            z = 0;
      } else if (j == 300) {
            t = 1530;
            z = 0;
      } else if (j == 320) {
            t = 1630;
            z = 0;
      } else if (j == 340) {
            t = 1730;
            z = 0;
      } else if (j == 360) {
            t = 1830;
            z = 0;
      } else if (j == 380) {
            t = 1930;
```

```
                z = 0;
            } else if (j == 400) {
                t = 2030;
                z = 0;
            } else if (j == 420) {
                t = 2130;
                z = 0;
            } else if (j == 440) {
                t = 2230;
                z = 0;
            } else if (j == 460) {
                t = 2330;
                z = 0;
            }

            TT[j] = T[t];
            vv[j] = v[z];
            pp[j] = p[t][z];
            ss[j] = s[t][z];
            hh[j] = h[t][z];
            cvv[j] = cv[t][z];
            cpp[j] = cp[t][z];
            kpvv[j] = kpv[t][z];
            ktvv[j] = ktv[t][z];
            kptt[j] = kpt[t][z];
            gg[j] = g[t][z];
            ZZ[j] = Z[t][z];
            ataxx[j] = atax[t][z];
            cpidd[j]=cpid[t];
            z = z + 62;
        }

// apothikeush pinakwn elegxou se arxeia excel

        ExcelFile file = new ExcelFile();
        file.addColumn("Tcm", Tcm2);
        file.addColumn("pcm", pcm2);
        file.addColumn("vcm", vcm2);
        file.addColumn("Vcm", Vcm2);
        file.addColumn("T", TT);
        file.addColumn("v", vv);
        file.addColumn("p", pp);
        file.addColumn("s", ss);
        file.addColumn("h", hh);
        file.addColumn("cv", cvv);
        file.addColumn("cp", cpp);
        file.addColumn("kpv", kpvv);
        file.addColumn("ktv", ktvv);
        file.addColumn("kpt", kptt);
        file.addColumn("g", gg);
        file.addColumn("Z", ZZ);
        file.addColumn("atax", ataxx);
        file.addColumn("cpideal", cpidd);
        file.export("PRNuevo");/**/
```

```
// apothikeush pinakwn 2ou elegxou se arxeia Excel gia ta megethi cv, cp, g,
// kpv, a pou parousiazoun megalo sfalma gia th parastash kai ton upologismo
// sfalmatwn

            ExcelFile file0 = new ExcelFile();
            file0.addColumn("cv", cv[30]);
            file0.addColumn("cv", cv[130]);
            file0.addColumn("cv", cv[230]);
            file0.addColumn("cv", cv[330]);
            file0.addColumn("cv", cv[430]);
            file0.addColumn("cv", cv[530]);
            file0.addColumn("cv", cv[630]);
            file0.addColumn("cv", cv[730]);
            file0.addColumn("cv", cv[830]);
            file0.addColumn("cv", cv[930]);
            file0.addColumn("cv", cv[1030]);
            file0.addColumn("cv", cv[1130]);
            file0.addColumn("cv", cv[1230]);
            file0.addColumn("cv", cv[1330]);
            file0.addColumn("cv", cv[1430]);
            file0.addColumn("cv", cv[1530]);
            file0.addColumn("cv", cv[1630]);
            file0.addColumn("cv", cv[1730]);
            file0.addColumn("cv", cv[1830]);
            file0.addColumn("cv", cv[1930]);
            file0.addColumn("cv", cv[2030]);
            file0.addColumn("cv", cv[2130]);
            file0.addColumn("cv", cv[2230]);
            file0.addColumn("cv", cv[2330]);
            file0.export("cvsfalma");

            ExcelFile file00 = new ExcelFile();
            file00.addColumn("cp", cp[30]);
            file00.addColumn("cp", cp[130]);
            file00.addColumn("cp", cp[230]);
            file00.addColumn("cp", cp[330]);
            file00.addColumn("cp", cp[430]);
            file00.addColumn("cp", cp[530]);
            file00.addColumn("cp", cp[630]);
            file00.addColumn("cp", cp[730]);
            file00.addColumn("cp", cp[830]);
            file00.addColumn("cp", cp[930]);
            file00.addColumn("cp", cp[1030]);
            file00.addColumn("cp", cp[1130]);
            file00.addColumn("cp", cp[1230]);
            file00.addColumn("cp", cp[1330]);
            file00.addColumn("cp", cp[1430]);
            file00.addColumn("cp", cp[1530]);
            file00.addColumn("cp", cp[1630]);
            file00.addColumn("cp", cp[1730]);
            file00.addColumn("cp", cp[1830]);
            file00.addColumn("cp", cp[1930]);
            file00.addColumn("cp", cp[2030]);
```

```
file00.addColumn("cp", cp[2130]);
file00.addColumn("cp", cp[2230]);
file00.addColumn("cp", cp[2330]);
file00.export("cpsfalma");

ExcelFile fileg = new ExcelFile();
fileg.addColumn("g", g[30]);
fileg.addColumn("g", g[130]);
fileg.addColumn("g", g[230]);
fileg.addColumn("g", g[330]);
fileg.addColumn("g", g[430]);
fileg.addColumn("g", g[530]);
fileg.addColumn("g", g[630]);
fileg.addColumn("g", g[730]);
fileg.addColumn("g", g[830]);
fileg.addColumn("g", g[930]);
fileg.addColumn("g", g[1030]);
fileg.addColumn("g", g[1130]);
fileg.addColumn("g", g[1230]);
fileg.addColumn("g", g[1330]);
fileg.addColumn("g", g[1430]);
fileg.addColumn("g", g[1530]);
fileg.addColumn("g", g[1630]);
fileg.addColumn("g", g[1730]);
fileg.addColumn("g", g[1830]);
fileg.addColumn("g", g[1930]);
fileg.addColumn("g", g[2030]);
fileg.addColumn("g", g[2130]);
fileg.addColumn("g", g[2230]);
fileg.addColumn("g", g[2330]);
fileg.export("gsfalma");

ExcelFile filek = new ExcelFile();
filek.addColumn("k", kpv[30]);
filek.addColumn("k", kpv[130]);
filek.addColumn("k", kpv[230]);
filek.addColumn("k", kpv[330]);
filek.addColumn("k", kpv[430]);
filek.addColumn("k", kpv[530]);
filek.addColumn("k", kpv[630]);
filek.addColumn("k", kpv[730]);
filek.addColumn("k", kpv[830]);
filek.addColumn("k", kpv[930]);
filek.addColumn("k", kpv[1030]);
filek.addColumn("k", kpv[1130]);
filek.addColumn("k", kpv[1230]);
filek.addColumn("k", kpv[1330]);
filek.addColumn("k", kpv[1430]);
filek.addColumn("k", kpv[1530]);
filek.addColumn("k", kpv[1630]);
filek.addColumn("k", kpv[1730]);
filek.addColumn("k", kpv[1830]);
filek.addColumn("k", kpv[1930]);
filek.addColumn("k", kpv[2030]);
```

```
            filek.addColumn("k", kpv[2130]);
            filek.addColumn("k", kpv[2230]);
            filek.addColumn("k", kpv[2330]);
            filek.export("kpvsfalma");


            ExcelFile filea = new ExcelFile();
            filea.addColumn("a", atax[30]);
            filea.addColumn("a", atax[130]);
            filea.addColumn("a", atax[230]);
            filea.addColumn("a", atax[330]);
            filea.addColumn("a", atax[430]);
            filea.addColumn("a", atax[530]);
            filea.addColumn("a", atax[630]);
            filea.addColumn("a", atax[730]);
            filea.addColumn("a", atax[830]);
            filea.addColumn("a", atax[930]);
            filea.addColumn("a", atax[1030]);
            filea.addColumn("a", atax[1130]);
            filea.addColumn("a", atax[1230]);
            filea.addColumn("a", atax[1330]);
            filea.addColumn("a", atax[1430]);
            filea.addColumn("a", atax[1530]);
            filea.addColumn("a", atax[1630]);
            filea.addColumn("a", atax[1730]);
            filea.addColumn("a", atax[1830]);
            filea.addColumn("a", atax[1930]);
            filea.addColumn("a", atax[2030]);
            filea.addColumn("a", atax[2130]);
            filea.addColumn("a", atax[2230]);
            filea.addColumn("a", atax[2330]);
            filea.export("ataxsfalma");

// apothikeush pinakwn tou Z se arxeia Excel gia th grafikh parastash tou

            ExcelFile file1 = new ExcelFile();
            file1.addColumn("Tr", Trm[162]);// Tr=0.7
            file1.addColumn("pr", prm[162]);
            file1.addColumn("Z", Z[162]);

            file1.addColumn("Tr", Trm[345]);// Tr=0.75
            file1.addColumn("pr", prm[345]);
            file1.addColumn("Z", Z[345]);


            file1.addColumn("Tr", Trm[528]);// Tr=0.8
            file1.addColumn("pr", prm[528]);
            file1.addColumn("Z", Z[528]);

            file1.addColumn("Tr", Trm[895]);// Tr=0.9
            file1.addColumn("pr", prm[895]);
            file1.addColumn("Z", Z[895]);
```

```
            file1.addColumn("Tr", Trm[1261]);// Tr=1
            file1.addColumn("pr", prm[1261]);
            file1.addColumn("Z", Z[1261]);

            file1.addColumn("Tr", Trm[1627]);// Tr=1.1
            file1.addColumn("pr", prm[1627]);
            file1.addColumn("Z", Z[1627]);

            file1.addColumn("Tr", Trm[1993]);// Tr=1.2
            file1.addColumn("pr", prm[1993]);
            file1.addColumn("Z", Z[1993]);

            file1.addColumn("Tr", Trm[2360]);// Tr=1.3
            file1.addColumn("pr", prm[2360]);
            file1.addColumn("Z", Z[2360]);
            file1.export("DiagrammaZNuevo");

// entolh tupwshs sthn othonh xrhsimopoumnehs mnhmhs RAM

      mem();

      }

// dhmiourgia entolhs tupwshs xrhsimopoioumenhs mnhmhs RAM

      public static void mem() {

            long max = Runtime.getRuntime().maxMemory() / 1000000;
            long total = Runtime.getRuntime().totalMemory() / 1000000;
            long used = total - Runtime.getRuntime().freeMemory() / 1000000;
            System.out.println(used + "/" + total + "  " + max);

      }

// dhmiourgia sunarthshs upologismou p koresmenou aeriou

      public static double psat(double T) {

            double psat = 5.69458483406365 * Math.pow(10, -10.0) *
                        Math.pow(T, 6.0);
            psat = psat - 9.74950338883352 * Math.pow(10, -7.0) *
                        Math.pow(T, 5.0);
            psat = psat + 6.97039451880329 * Math.pow(10, -4.0) *
                        Math.pow(T, 4.0);
            psat = psat - 2.65021689221286 * Math.pow(10, -1.0) *
                        Math.pow(T, 3.0);
            psat = psat + 56.4714595802326 * Math.pow(T, 2.0);
            psat = psat - 6.39650290891730 * Math.pow(10, 3.0) * T;
            psat = psat + 3.01055282943808 * Math.pow(10, 5.0);

            return psat;

      }
```

```
// dhmiourgia sunarthshs upologismou v koresmenou aeriou

	public static double vsat1(double T) {

		double vsat = 2.63572780048964 * Math.pow(10, -12.0)*
				 Math.pow(T, 6.0);
		vsat = vsat - 4.48159674225455 * Math.pow(10, -9.0) *
				Math.pow(T, 5.0);
		vsat = vsat + 3.18157053817254 * Math.pow(10, -6.0) *
				Math.pow(T, 4.0);
		vsat = vsat - 1.20756541039049 * Math.pow(10, -3.0) *
				Math.pow(T, 3.0);

		vsat = vsat + 2.58571759571226 * Math.pow(10, -1.0) *
				Math.pow(T, 2.0);
		vsat = vsat - 29.6351741372027 * T;
		vsat = vsat + 1421.50529203835;

		return vsat;

	}

	public static double vsat2(double T) {

		double vsat = 1.74044211304108 * Math.pow(10, -14.0)*
				Math.pow(T, 6.0);
		vsat = vsat - 4.21168350304823 * Math.pow(10, -11.0) *
				Math.pow(T, 5.0);
		vsat = vsat + 4.15785301486820 * Math.pow(10, -8.0) *
				Math.pow(T, 4.0);
		vsat = vsat - 2.15890107344439 * Math.pow(10, -5.0) *
				Math.pow(T, 3.0);
		vsat = vsat + 6.25028354696890 * Math.pow(10, -3.0) *
				Math.pow(T, 2.0);
		vsat = vsat - 0.960589224331175 * T;
		vsat = vsat + 61.4566526436002;

		return vsat;

	}

// dhmiourgia sunarthshs metratophs monadwn ogkou apo m3/kg se cm3/mol

	public static double V (double v, double Mol) {

		double V = 1000 * Mol * v;

		return V;

	}
```

```
// dhmiourgia sunarthshs upologismou anhgmenhs thermokrasias Tr gia kathe
// sustatiko

	public static double[] tr(double T, double[] Tc) {

		double[] Tr = new double[3];
		for (int j = 0; j < 3; j++) {
			Tr[j] = T / Tc[j];
		}

		return Tr;

	}


// dhmiourgia sunarthshs upologismou anhgmenhs thermokrasias Tr migmatos

	public static double trm(double T, double Tcm2) {

		double Trm = T / Tcm2;

		return Trm;

	}


// dhmiourgia sunarthshs upologismou anhgmenou ogkou vr migmatos (cm3/mol)

	public static double vrm(double v, double vcm2) {

		double vrm = v / vcm2;

		return vrm;

	}


// dhmiourgia sunarthshs upologismou anhgmenou ogkou vr migmatos (m3/kg)

	public static double Vrm(double V, double Vcm2) {

		double Vrm = V / Vcm2;

		return Vrm;

	}
```

```
// dhmiourgia sunarthshs upologismou statheras a (kPa cm6/mol2)
// stathera eksartwmnh apo th thermokrasia

      public static double[] a(double[] Tr, double[] ac, double[] a1,
                  double[] a2, double[] a3) {

            double[] a = new double[3];

            for (int j = 0; j < 3; j++) {
                  a[j] = 1.0 - Math.sqrt(Tr[j]);
                  a[j] = 1.0 + a1[j] * a[j] + a2[j] * Math.pow(a[j], 2.0)
                        + a3[j] * Math.pow(a[j], 3.0);
                  a[j] = ac[j] * Math.pow(a[j], 2.0);
            }

            return a;

      }

// dhmiourgia sunarthshs upologismou paragonta diortwshs kij
// paragwntas eksartwmenos apo th thermokrasia

      public static double[][] k(double T) {

            double[][] k = new double[3][3];

            for (int j = 0; j < 3; j++) {
                  for (int m = 0; m < 3; m++) {

                        if ((j == 0 && m == 1) || (j == 1 && m == 0)) {
                              k[j][m] = 0.0;
                        } else if ((j == 0 && m == 2) || (j == 2 && m == 0)){
                              k[j][m] = 0.0;
                        } else if ((j == 1 && m == 2) || (j == 2 && m == 1)){
                              k[j][m] = 0.0;
                        } else {
                              k[j][m] = 0.0;
                        }
                  }
            }

            return k;

      }

// dhmiourgia sunarthshs upologismou paragonta diortwshs k123
// paragwntas eksartwmenos apo th thermokrasia

      public static double k123(double T) {

            double k123 = 0.0;

            return k123;
      }
```

```
// dhmiourgia sunarthshs upologismou statheras aij
// stathera eksartwmenh apo th thermokrasia

	public static double[][] aIJ(double[] a, double[][] k) {

		double[][] aIJ = new double[3][3];

		for (int j = 0; j < 3; j++) {

			for (int m = 0; m < 3; m++) {

				aIJ[j][m] = Math.sqrt(a[j] * a[m]) * (1.0 - k[j][m]);
			}
		}

		return aIJ;

	}


// dhmiourgia sunarthshs upologismou statheras a123
// stathera eksartwmenh apo th thermokrasia

	public static double a123(double k123, double[] a) {

		double a123 = k123 * Math.pow(a[0] * a[1] * a[2], (1 / 3.0));

		return a123;

	}


// dhmiourgia sunarthshs upologismou statheras am
// stathera eksartwmenh apo th thermokrasia

	public static double am(double[][] aIJ, double a123, double[] x) {

		double am = 0;

		for (int m = 0; m < 3; m++) {

			for (int j = 0; j < 3; j++) {

				am = x[j] * x[m] * aIJ[j][m] + am;

			}
		}
		am = am + 3.0 * a123 * x[0] * x[1] * x[2];

		return am;

	}
```

```
// dhmiourgia sunarthshs upologismou paragonta diortwshs lij

	public static double[][] l() {

		double[][] l = new double[3][3];

		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {
				if ((j == 0 & m == 1) || (j == 1 && m == 0)) {
					l[j][m] = 0.0;
				} else if ((j == 0 & m == 2) || (j == 2 && m == 0)) {
					l[j][m] = 0.0;
				} else if ((j == 1 & m == 2) || (j == 2 && m == 1)) {
					l[j][m] = 0.0;
				} else {
					l[j][m] = 0;
				}
			}
		}

		return l;

	}

// dhmiourgia sunarthshs upologismou statheras bij

	public static double[][] BIJ(double[] b, double[][] l) {

		double[][] BIJ = new double[3][3];
		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {
				BIJ[j][m] = (b[j] + b[m]) / 2.0;
				BIJ[j][m] = BIJ[j][m] * (1.0 - l[j][m]);
			}
		}

		return BIJ;

	}

// dhmiourgia sunarthshs upologismou statheras bm

	public static double bm(double[][] B, double[] x) {

		double bm = 0;
		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {
				bm = x[j] * x[m] * B[j][m] + bm;
			}
		}

		return bm;

	}
```

```
// dhmiourgia sunarthshs upologismou pieshs p (kPa)
// megethos eksartwmeno apo thermokrasia kai ogko

      public static double p(double am, double bm, double Rmol, double T,
            double V){

            double p = 1000 * Rmol * T / (V - bm)
                  - (am / (Math.pow(V, 2) + 2 * V * bm - Math.pow(bm, 2)));

            return p;

      }

// dhmiourgia sunarthshs upologismou anhgmenhs pieshs migmatos prm
// megethos eksartwmeno apo thermokrasia kai ogko

      public static double prm(double p, double pcm) {

            double prm = p / pcm;

            return prm;

      }

// dhmiourgia sunarthshs upologismou statheras dadt (paragwgos a)
// megethos eksartwmeno apo th thermokrasia

      public static double[] dadt(double[] Tr, double[] ac, double[] a1,
                  double[] a2, double[] a3, double[] Tc) {

            double[] dadt = new double[3];
            double[] B = new double[3];

            for (int j = 0; j < 3; j++) {
                  dadt[j] = 1.0 - Math.sqrt(Tr[j]);
                  dadt[j] = 1.0 + a1[j] * dadt[j]
                        + a2[j] * Math.pow(dadt[j], 2.0)
                        + a3[j] * Math.pow(dadt[j], 3.0);

                  B[j] = 1.0 - Math.sqrt(Tr[j]);
                  B[j] = a1[j] + 2.0 * a2[j] * B[j]
                        + 3.0 * a3[j]* Math.pow(B[j], 2.0);
                  B[j] = -(1.0 / (Math.sqrt(Tr[j]) * Tc[j])) * B[j];

                  dadt[j] = ac[j] * dadt[j] * B[j];

            }

            return dadt;

      }
```

```
// dhmiourgia sunarthshs upologismou paragonta diortwshs dkdt (paragwgos k)
// paragwntas eksartwmenos apo th thermokrasia

	public static double[][] dkdt() {

		double[][] dkdt = new double[3][3];

		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {
				if ((j == 0 && m == 1) || (j == 1 && m == 0)) {
					dkdt[j][m] = 0.0;
				} else if ((j == 0 && m == 2) || (j == 2 && m == 0)){
					dkdt[j][m] = 0.0;
				} else if ((j == 1 && m == 2) || (j == 2 && m == 1)){
					dkdt[j][m] = 0.0;
				} else {
					dkdt[j][m] = 0.0;
				}
			}
		}

		return dkdt;

	}


// dhmiourgia sunarthshs upologismou statheras da123dt (paragwgos a123)
// megethos eksartwmeno apo th thermokrasia

	public static double da123dt(double k123, double[] a, double dk123dt,
			double[] dadt) {

		double B = dadt[0] * a[1] * a[2] + dadt[1] * a[0] * a[2]
				+ dadt[2]* a[1] * a[0];
		B = B / (Math.pow((a[0] * a[1] * a[2]), (2.0 / 3.0)));

		double da123dt = Math.pow((a[0] * a[1] * a[2]), (1.0 / 3.0));
		da123dt = da123dt * dk123dt + k123 * B;

		return da123dt;

	}

// dhmiourgia sunarthshs upologismou statheras daijdt (paragwgos aij)
// megethos eksartwmeno apo th thermokrasia

	public static double[][] daIJdt(double[] a, double[][] k,
			double[] dadt,double[][] dkdt) {

		double[][] daIJdt = new double[3][3];

		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {
				daIJdt[j][m] = 0.5 * Math.sqrt(a[m] / a[j]) * dadt[j]
```

```
                                    * (1.0 - k[j][m]);
                        daIJdt[j][m] = 0.5 * Math.sqrt(a[j] / a[m]) * dadt[m]
                                    * (1.0 - k[j][m]) + daIJdt[j][m];
                        daIJdt[j][m] = daIJdt[j][m] - Math.sqrt(a[j] * a[m])
                                    * dkdt[j][m];
                  }
            }

            return daIJdt;
      }

// dhmiourgia sunarthshs upologismou statheras damdt (paragwgos am)
// megethos eksartwmeno apo th thermokrasia

      public static double damdt(double[][] daIJdt, double da123dt,
                  double[] x) {

            double damdt = 0;
            for (int m = 0; m < 3; m++) {
                  for (int j = 0; j < 3; j++) {
                        damdt = x[j] * x[m] * daIJdt[j][m] + damdt;
                  }
            }
            damdt = damdt + 3.0 * da123dt * x[0] * x[1] * x[2];

            return damdt;
      }
// dhmiourgia sunarthshs upologismou paragwgou pieshs ws pros thermokrasia
// megethos eksartwmeno apo th thermokrasia kai ton ogko (kpa/K)

      public static double dpdt(double damdt, double bm, double Rmol,
                  double V) {

            double dpdt = 1000 * Rmol / (V - bm)
                        - (damdt / (Math.pow(V, 2.0)
                        + 2 * V * bm - Math.pow(bm, 2.0)));

            return dpdt;

      }

// dhmiourgia sunarthshs upologismou paragwgou pieshs ws pros ogko
// megethos eksartwmeno apo th thermokrasia kai ton ogko (kPa/(cm3/mol))

public static double dpdv(double am, double bm, double Rmol, double T,
                  double V) {

            double dpdv = Math.pow(V, 2) + 2 * V * bm - Math.pow(bm, 2);
            dpdv = (-1000 * Rmol * T / (Math.pow((V - bm), 2.0)))
                        + ((am * 2.0 * (V + bm)) / (Math.pow(dpdv, 2.0)));

            return dpdv;

      }
```

```
// dhmiourgia sunarthshs upologismou paragwgou entropias s (kJ/kgK)
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double s(double damdt, double bm, double Rmol, double R,
                  double Mol, double T, double To, double V, double so,
                  double po,double A, double B, double C, double D) {

            double r = 1 / V;

            double s = Math.log(Math.abs((-bm * r + 1.0 - Math.sqrt(2.0))
                        / (-bm * r + 1.0 + Math.sqrt(2.0))));
            s = s - Math.log(Math.abs((1.0 - Math.sqrt(2.0))
                        / (1.0 + Math.sqrt(2.0))));
            s = damdt * s / (bm * Math.sqrt(8.0));
            s = Rmol * Math.log(1.0 - r * bm) + s;
            s = s / (1000 * Mol);
            s = -R * Math.log(r * 1000 * Rmol * T / po) + s;
            s = -A * Math.log(To) - B * To - 0.5 * C * Math.pow(To, 2.0)
                        - (1/3.0) * D * Math.pow(To, 3.0) + s;
            s = A * Math.log(T) + B * T + 0.5 * C * Math.pow(T, 2.0)
                        + (1/3.0) * D * Math.pow(T, 3.0) + so + s;

            return s;

      }

// dhmiourgia sunarthshs upologismou paragwgou enthalpias h (kJ/kg)
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double h(double damdt, double am, double bm, double R,
                  double Mol, double T, double To, double V, double Uo, double
                  p,double A, double B, double C, double D) {

            double r = 1 / V;

            double h = Math.log(Math.abs((-bm * r + 1.0 - Math.sqrt(2.0))
                        / (-bm * r + 1.0 + Math.sqrt(2.0))));
            h = h - Math.log(Math.abs((1.0 - Math.sqrt(2.0))
                        / (1.0 + Math.sqrt(2.0))));
            h = (-am + T * damdt) * h / ((bm * Math.sqrt(8.0)));
            h = p / r + h;
            h = h / (1000 * Mol);
            h = -R * (T - To) + h;
            h = -A * To - 0.5 * B * Math.pow(To, 2.0) - (1 / 3.0) * C
                  * Math.pow(To, 3.0) - 0.25 * D * Math.pow(To, 4.0) +h;
            h = A * T + 0.5 * B * Math.pow(T, 2.0) + (1 / 3.0) * C
                  * Math.pow(T, 3.0) + 0.25 * D * Math.pow(T, 4.0) + h;
            h = Uo + h;

            return h;

      }
```

```
// dhmiourgia sunarthshs upologismou ths 2hs paragwgou ths statheras a
// megethos eksartwmeno apo th thermokrasia

      public static double[] d2adt2(double[] Tr, double[] ac, double[] a1,
                  double[] a2, double[] a3, double[] Tc) {

            double[] d2adt2 = new double[3];
            double[] B = new double[3];
            double[] C = new double[3];
            double[] D = new double[3];
            for (int j = 0; j < 3; j++) {
                  d2adt2[j] = 1.0 - Math.sqrt(Tr[j]);
                  d2adt2[j] = 1.0 + a1[j] * d2adt2[j]
                        + a2[j] * Math.pow(d2adt2[j], 2.0)
                        + a3[j] * Math.pow(d2adt2[j], 3.0);
                  B[j] = 1.0 - Math.sqrt(Tr[j]);
                  B[j] = a1[j] + 2.0 * a2[j] * B[j]
                        + 3.0 * a3[j] * Math.pow(B[j], 2.0);
                  C[j] = ac[j] / (2.0 * Tr[j] * Math.pow(Tc[j], 2.0));
                  D[j] = 1.0 - Math.sqrt(Tr[j]);
                  D[j] = 6.0 * a3[j] * D[j] + 2.0 * a2[j];
                  d2adt2[j] = C[j]* (d2adt2[j] * B[j] / (Math.sqrt(Tr[j]))
                        + Math.pow(B[j], 2.0) + d2adt2[j] * D[j]);
            }

            return d2adt2;

      }

// dhmiourgia sunarthshs upologismou ths 2hs paragwgou ths statheras a123
// megethos eksartwmeno apo th thermokrasia

      public static double d2a123dt2(double k123, double[] a, double dk123dt,
                  double[] dadt, double[] d2adt2) {

            double A = Math.pow((a[0] * a[1] * a[2]), (1.0 / 3.0));
            double B = dadt[0] * a[1] * a[2] + dadt[1] * a[0] * a[2]
                  + dadt[2]* a[1] * a[0];
            double C = B / (3.0 * Math.pow(A, 2.0));
            double D = d2adt2[0] * a[1] * a[2]
                  + d2adt2[1] * a[0] * a[2] + d2adt2[2] * a[1] * a[0]
                  + 2.0 * (dadt[0] * dadt[1] * a[2]
                  + dadt[2] * dadt[1] * a[0] + dadt[0] * dadt[2] * a[1]);
            D = D / (3.0 * Math.pow(A, 2.0));
            D = -(2.0 / 9.0) * Math.pow(B, 2.0) / Math.pow(A, 5.0) + D;

            double d2a123dt2 = 2.0 * C * dk123dt + k123 * D;

            return d2a123dt2;

      }
```

```
// dhmiourgia sunarthshs upologismou ths 2hs paragwgou ths statheras aij
// megethos eksartwmeno apo th thermokrasia

      public static double[][] d2aIJdt2(double[] a, double[][] k,
                  double[] dadt, double[][] dkdt, double[] d2adt2) {

            double[][] d2aIJdt2 = new double[3][3];

            for (int j = 0; j < 3; j++) {
                  for (int m = 0; m < 3; m++) {
                        d2aIJdt2[j][m] = -dkdt[j][m] * (a[j] * dadt[m]
                              + a[m] * dadt[j]) / Math.sqrt(a[j] * a[m]);

                        d2aIJdt2[j][m] = d2aIJdt2[j][m] - (1 - k[j][m])
                              * Math.pow(a[j] * dadt[m]
                              + a[m] * dadt[j], 2.0)
                              / (4 * Math.pow(Math.sqrt(a[j] * a[m]), 3.0));

                        d2aIJdt2[j][m] = d2aIJdt2[j][m] + (1 - k[j][m])
                              * (a[j] * d2adt2[m] + a[m] * d2adt2[j]
                              + 2.0 * dadt[j] * dadt[m])
                              / (2 * (Math.sqrt(a[j] * a[m])));
                  }
            }

            return d2aIJdt2;

      }



// dhmiourgia sunarthshs upologismou ths 2hs paragwgou ths statheras am
// megethos eksartwmeno apo th thermokrasia

      public static double d2amdt2(double[][] d2aIJdt2, double d2a123dt2,
                  double[] x) {

            double d2amdt2 = 0;
            for (int m = 0; m < 3; m++) {
                  for (int j = 0; j < 3; j++) {
                        d2amdt2 = x[j] * x[m] * d2aIJdt2[j][m] + d2amdt2;
                  }
            }
            d2amdt2 = d2amdt2 + 3.0 * d2a123dt2 * x[0] * x[1] * x[2];

            return d2amdt2;

      }
```

```
// dhmiourgia sunarthshs upologismou paragwgou eidikhs thermoxwrhtikothtas
// upo stathero ogko cv (kJ/kg K)
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double cv(double d2amdt2, double bm, double R,
                  double Mol, double T, double V, double A, double B,
                  double C, double D) {

            double r = 1 / V;
            double cv = Math.log(Math.abs((-bm * r + 1.0 - Math.sqrt(2.0))
                        / (-bm * r + 1.0 + Math.sqrt(2.0))));
            cv = cv - Math.log(Math.abs((1.0 - Math.sqrt(2.0))
                        / (1.0 + Math.sqrt(2.0))));
            cv = T * d2amdt2 * cv / (1000 * Mol * bm * Math.sqrt(8.0));
            cv = cv - R;
            cv = cv + A + B * T + C * Math.pow(T, 2.0)
                        + D * Math.pow(T, 3.0);
            return cv;

      }

// dhmiourgia sunarthshs upologismou paragwgou eidikhs thermoxwrhtikothtas
// upo statherh piesh cp (kJ/kg K)
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double cp(double cv, double Mol, double T, double dpdt,
                  double dpdv) {

            double cp = cv - T * Math.pow(dpdt, 2.0) / (dpdv * 1000 * Mol);
            return cp;

      }

// dhmiourgia sunarthshs upologismou isentropikou suntelesth kpv
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double kpv(double V, double p, double cv, double cp,
                  double dpdv) {

            double kpv = -V * cp * dpdv / (p * cv);
            return kpv;

      }

// dhmiourgia sunarthshs upologismou isentropikou suntelesth ktv
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double ktv(double V, double cv, double dpdt, double Mol){

            double ktv = 1.0 + V * dpdt / (cv * 1000.0 * Mol);

            return ktv;

      }
```

```
// dhmiourgia sunarthshs upologismou isentropikou suntelesth kpt
// megethos eksartwmeno apo th thermokrasia kai ton ogko


      public static double kpt(double T, double p, double cv, double cp,
                  double dpdt) {

            double kpt = T * dpdt / (T * dpdt + p * (cv / cp - 1.0));

            return kpt;

      }


// dhmiourgia sunarthshs upologismou logou eidikvn thermoxwrhtikothtwn g
// megethos eksartwmeno apo th thermokrasia kai ton ogko


      public static double g(double cv, double cp) {

            double g = cp / cv;

            return g;

      }


// dhmiourgia sunarthshs upologismou suntelesth sumpiestothtas Z
// megethos eksartwmeno apo th thermokrasia kai ton ogko


      public static double Z(double p, double V, double T, double Rmol) {

            double Z = p * V / (1000 * Rmol * T);

            return Z;

      }


// dhmiourgia sunarthshs upologismou taxuthtas hxou a (m/s)
// megethos eksartwmeno apo th thermokrasia kai ton ogko


      public static double atax(double V, double cv, double cp, double dpdv,
                  double Mol) {

            double atax = -Math.pow(V, 2.0) * cp * dpdv / (cv * Mol);
            atax = Math.sqrt(atax);

            return atax;

      }
```

```
// dhmiourgia sunarthshs upologismou statheras a gia th krisimh thermokrasia
// migmatos

	public static double[] A1(double Tcm, double[] Tc, double[] w,
			double[] pc, double Rmol) {//
		double[] A1 = new double[3];
		double[] a1 = new double[3];
		double[] ac = new double[3];
		for (int j = 0; j < 3; j++) {
			A1[j] = 1.0 - Math.sqrt(Tcm / Tc[j]);
			a1[j] = 0.37464 + 1.54226 * w[j]
				- 0.26992 * Math.pow(w[j], 2.0);
			A1[j] = 1.0 + a1[j] * A1[j];
			ac[j] = 0.457235 * Math.pow(Rmol, 2.0)
				* Math.pow(Tc[j], 2.0) / (0.000001 * pc[j]);
			A1[j] = ac[j] * Math.pow(A1[j], 2.0);
		}

		return A1;
	}

// dhmiourgia sunarthshs upologismou paragonta diortwshs k gia  th krisimh
// thermokrasia migmatos

	public static double[][] K(double Tcm) {
		double[][] K = new double[3][3];
		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {
				if ((j == 0 && m == 1) || (j == 1 && m == 0)) {
					K[j][m] = 0.0;
				} else if ((j == 0 && m == 2) || (j == 2 && m == 0)){
					K[j][m] = 0.0;
				} else if ((j == 1 && m == 2) || (j == 2 && m == 1)){
					K[j][m] = 0.0;
				} else {
					K[j][m] = 0.0;
				}
			}
		}

		return K;
	}

// dhmiourgia sunarthshs upologismou paragonta diortwshs k123 gia  th krisimh
// thermokrasia migmatos

	public static double K123(double Tcm) {

		double K123 = 0.0;

		return K123;

	}
```

```
// dhmiourgia sunarthshs upologismou paragonta diortwshs aij gia  th krisimh
// thermokrasia migmatos


      public static double[][] AIJ(double[] A1, double[][] K) {

            double[][] AIJ = new double[3][3];

            for (int j = 0; j < 3; j++) {
                  for (int m = 0; m < 3; m++) {
                        AIJ[j][m] = Math.sqrt(A1[j] * A1[m])
                              * (1.0 - K[j][m]);
                  }
            }

            return AIJ;

      }


// dhmiourgia sunarthshs upologismou statheras a123 gia  th krisimh
// thermokrasia migmatos


      public static double A123(double K123, double[] A1) {

            double A123 = K123 * Math.pow(A1[0] * A1[1] * A1[2], (1 / 3.0));

            return A123;

      }


// dhmiourgia sunarthshs upologismou statheras am gia  th krisimh
// thermokrasia migmatos


      public static double Am(double[][] AIJ, double A123, double[] x) {
            double Am = 0;
            for (int m = 0; m < 3; m++) {
                  for (int j = 0; j < 3; j++) {
                        Am = x[j] * x[m] * AIJ[j][m] + Am;
                  }
            }
            Am = Am + 3.0 * A123 * x[0] * x[1] * x[2];

            return Am;

      }
```

```
// dhmiourgia sunarthshs upologismou krisimhs thermokrasias symfwna me tis
// proteinomenes sxeseis gia thn katastatikh eksiswsh Peng-Robinson


      public static double Tcm1(double Am, double[] Tc, double[] pc,
                  double[] x, double Rmol) {

            double A = x[0] * Tc[0] / pc[0] + x[1] * Tc[1] / pc[1]
                  + x[2] * Tc[2] / pc[2];

            double Tcm1 = Am / (1000000.0 * 0.457235
                  * Math.pow(Rmol, 2.0) * A);

            return Tcm1;

      }


// dhmiourgia sunarthshs upologismou krisimhs pieshs symfwna me tis
// proteinomenes sxeseis gia thn katastatikh eksiswsh Peng-Robinson


      public static double pcm1(double[] Tc, double[] pc, double[] x,
                  double Tcm1) {

            double pcm1;
            double A = x[0] * Tc[0] / pc[0] + x[1] * Tc[1] / pc[1]
                  + x[2] * Tc[2] / pc[2];

            pcm1 = Tcm1 / A;

            return pcm1;

      }


// dhmiourgia sunarthshs upologismou krisimou ogkou symfwna me tis
// proteinomenes sxeseis gia thn katastatikh eksiswsh Peng-Robinson


      public static double Vcm1(double Tcm1, double pcm1, double Am,
                  double bm, double Rmol, double Vcm1Old) {

            double Vcm1 = (1000 * Rmol * Tcm1 / pcm1) - Am
                        / ((bm + Vcm1Old * ((Vcm1Old + bm)
                        / (Vcm1Old - bm))) * pcm1) + bm;

            return Vcm1;

      }
```

```
// dhmiourgia sunarthshs upologismou idanikhs eidikhs thermoxwrhtikothtas
// cpideal


	public static double cpideal(double T, double A, double B, double C,
			double D) {

		double cpid = A + B * T + C * Math.pow(T, 2.0)
			+ D * Math.pow(T, 3.0);

		return cpid;

	}


// dhmiourgia sunarthshs upologismou paragwnta Vij symfwna me tis
// proteinomenes sxeseis gia thn katastatikh eksiswsh Lee-Kesler


	public static double [][] Vcij (double [] Vc) {

		double[][] Vcij = new double[3][3];

		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {

				Vcij[j][m] =0.125 * Math.pow((Math.pow(Vc[j],(1/3.0))
					+ Math.pow(Vc[m],(1/3.0))), 3.0);
			}
		}

		return Vcij;

	}


// dhmiourgia sunarthshs upologismou krisimou ogkou symfwna me tis
// proteinomenes sxeseis gia thn katastatikh eksiswsh Lee-Kesler


	public static double  Vcm2 (double [][] Vcij, double [] x) {
		double Vcm2 = 0;
		for (int m = 0; m < 3; m++) {
			for (int j = 0; j < 3; j++) {
				Vcm2 =x[j] * x[m] * Vcij[j][m] + Vcm2;
			}
		}

		return Vcm2;

	}
```

```
// dhmiourgia sunarthshs upologismou paragwnta Tij symfwna me tis
// proteinomenes sxeseis gia thn katastatikh eksiswsh Lee-Kesler

      public static double [][] Tcij (double [] Tc) {

            double[][] Tcij = new double[3][3];

            for (int j = 0; j < 3; j++) {
                  for (int m = 0; m < 3; m++) {

                        Tcij[j][m] = Math.sqrt(Tc[j]* Tc[m]);
                  }
            }

            return Tcij;

      }

// dhmiourgia sunarthshs upologismou krisimhs thermokrasias symfwna me tis
// proteinomenes sxeseis gia thn katastatikh eksiswsh Lee-Kesler

      public static double  Tcm2 (double [][] Tcij, double [][] Vcij,
                  double Vcm2, double [] x) {

            double Tcm2 = 0;

            for (int m = 0; m < 3; m++) {
                  for (int j = 0; j < 3; j++) {
                        Tcm2 = x[j] * x[m] * Tcij[j][m]
                              * Math.pow(Vcij[j][m], 0.25)+ Tcm2;
                  }
            }

            Tcm2=Tcm2/(Math.pow(Vcm2, 0.25));

            return Tcm2;

      }

}
```

```
// dhmiourgia fakelou refrigeration (‘package’) opou apothikeuetai to arxeio

package refrigeration;

// fortwma to arxeio java (‘class’) gia tis apothikeuseis twn pinakwn sto
// Excel

import katharo.ExcelFile;

// dhmiourgia arxeio java (‘class’) to opoio tha upoligizei oles tis
// thermophusikes idiothtes

public class DiagrammataR152aR125R32 {

// dhmiourgia basikhs ‘method’ opou grafontai oles oi kentrikes entoles
// (dhmiourgia pinakwn, apothikeush timwn se aytous apo sunarthseis,
// ektupwseis ktl

public static void main(String[] args) {

// dhmiourgia pinaka piesewn gia thn ekponhsh twn diagrammatwn

            double [] p = new double[10];
            p[0]=200.0;
            p[1]=300.0;
            p[2]=400.0;
            p[3]=500.0;
            p[4]=700.0;
            p[5]=1000.0;
            p[6]=1500.0;
            p[7]=2000.0;
            p[8]=3000.0;
            p[9]=4000.0;

// dhmiourgia pinaka me ogkous koresmou gia tis antistoixes pieseis

            double [] vsat = new double[10];
            vsat[0]=0.15126;
            vsat[1]=0.10276;
            vsat[2]=0.077883;
            vsat[3]=0.062679;
            vsat[4]=0.044954;
            vsat[5]=0.031319;
            vsat[6]=0.020403;
            vsat[7]=0.014779;
            vsat[8]=0.0089292;
            vsat[9]=0.0057472;

// apothikeush se pinakes twn statherwn twn ousiwn

            double[] Tc = new double[3];        // krisimes thermokrasies
            Tc[0] = 386.44;
            Tc[1] = 339.45;
            Tc[2] = 351.55;
```

```
double[] pc = new double[3];        // krisimes pieseis
pc[0] = 4520.0;
pc[1] = 3630.6;
pc[2] = 5830.0;

double[] w = new double[3];         // krisimoi acentric factors
w[0] = 0.252851;
w[1] = 0.294334;
w[2] = 0.267817;

double Rmol = 8.31447215;           // pagosmia stathera aeriwn
                                    // se KJ/KmolK

double[] ac = new double[3];        //stahera ac ths Peng-Robinson
ac[0] = 0.457235 * Math.pow(Rmol, 2.0) * Math.pow(Tc[0], 2.0)
            / (0.000001 * pc[0]);
ac[1] = 0.457235 * Math.pow(Rmol, 2.0) * Math.pow(Tc[1], 2.0)
            / (0.000001 * pc[1]);
ac[2] = 0.457235 * Math.pow(Rmol, 2.0) * Math.pow(Tc[2], 2.0)
            / (0.000001 * pc[2]);

double[] a1 = new double[3];        //stahera a1 ths Peng-Robinson
a1[0] = 0.826430;
a1[1] = 0.824698;
a1[2] = 0.863418;

double[] a2 = new double[3];        //stahera a2 ths Peng-Robinson
a2[0] = 0.0;
a2[1] = 0.0;
a2[2] = 0.0;

double[] a3 = new double[3];        //stahera a3 ths Peng-Robinson
a3[0] = 0.0;
a3[1] = 0.0;
a3[2] = 0.0;

double[] b = new double[3];         //stahera b ths Peng-Robinson
b[0] = 0.077796 * Rmol * Tc[0] / (0.001 * pc[0]);
b[1] = 0.077796 * Rmol * Tc[1] / (0.001 * pc[1]);
b[2] = 0.077796 * Rmol * Tc[2] / (0.001 * pc[2]);

double[] M = new double[3];         // moriaka barh
M[0] = 66.050;
M[1] = 120.0215;
M[2] = 52.035;

double[] n = new double[3];         // sustash kata maza
n[0] = 0.48;
n[1] = 0.18;
n[2] = 0.34;

double so = 1.675 + 0.456263644;    // times shmeiou anaforas
double po = 251.325;
double To = 273.15;
```

```
            double Uo = 292.5 + 162.804172;
            double A = 0.482584348;             // statheres cpideal
            double B = 6.2991266 * 0.0001;
            double C = 4.0487984 * Math.pow(10,-6.0);
            double D = -4.24878898 * Math.pow(10,-9.0);

// upologismos sustashs kata mol

            double[] x = new double[3];
            x[0] = n[0] / (n[0] + n[1] * M[0] / M[1] + n[2] * M[0] /M[2]);
            x[1] = n[1] / (n[1] + n[2] * M[1] / M[2] + n[0] * M[1] /M[0]);
            x[2] = n[2] / (n[2] + n[0] * M[2] / M[0] + n[1] * M[2] /M[1]);

// upologismos sustashs MB,acentric factor, statheras aeriou migmatos

            double Mol = x[0] * M[0] + x[1] * M[1] + x[2] * M[2];
            double wm = w[0] * x[0] + w[1] * x[1] + w[2] * x[2];
            double R = Rmol / Mol;              // se KJ/kgK

// dhmiourgia pinakwn opou tha apothikeuountai oi times twn thermodunamikwn
// megethwn

            double[] T = new double[2410];
            double[][] v = new double[p.length][T.length]; // m3 / kg
            double[][] V = new double[p.length][T.length]; // cm3 / mol
            double[][] Tr = new double[T.length][3];
            double[][] a = new double[T.length][3];
            double[][][] k = new double[T.length][3][3];
            double[] k123 = new double[T.length];
            double[][][] aIJ = new double[T.length][3][3];
            double[] a123 = new double[T.length];
            double[] am = new double[T.length];
            double[][] l = l();
            double[][] BIJ = BIJ(b, l);
            double bm = bm(BIJ, x);
            double[][] dadt = new double[T.length][3];
            double[][] dkdt = dkdt();
            double dk123dt = 0.0;
            double[] da123dt = new double[T.length];
            double[][][] daIJdt = new double[T.length][3][3];
            double[] damdt = new double[T.length];
            double[][] dpdt = new double[p.length][T.length];
            double[][] dpdv = new double[p.length][T.length];
            double[][] s = new double[p.length][T.length];
            double[][] h = new double[p.length][T.length];
            double[][] d2adt2 = new double[T.length][3];
            double[] d2a123dt2 = new double[T.length];
            double[][][] d2aIJdt2 = new double[T.length][3][3];
            double[] d2amdt2 = new double[T.length];
            double[][] cv = new double[p.length][T.length];
            double[][] cp = new double[p.length][T.length];
            double[][] kpv = new double[p.length][T.length];
            double[][] ktv = new double[p.length][T.length];
            double[][] kpt = new double[p.length][T.length];
```

```
            double[][] g = new double[p.length][T.length];
            double[][] Z = new double[p.length][T.length];
            double[][] atax = new double[p.length][T.length];

// apothikeush stous antistoixous pinakes twn timwn twn megethwn pou
// eksartountai mono apo th T kalwmtas thn katallhlh sunarthsh me to
// katallhlo stoixeio eisodou

            for (int i = 0; i < 2410; i++) {
                  T[i] = 240.15 + 0.1 * i;
                  Tr[i] = tr(T[i], Tc);
                  a[i] = a(Tr[i], ac, a1, a2, a3);
                  k[i] = k(T[i]);
                  k123[i] = k123(T[i]);
                  aIJ[i] = aIJ(a[i], k[i]);
                  a123[i] = a123(k123[i], a[i]);
                  am[i] = am(aIJ[i], a123[i], x);
                  dadt[i] = dadt(Tr[i], ac, a1, a2, a3, Tc);
                  da123dt[i] = da123dt(k123[i], a[i], dk123dt, dadt[i]);
                  daIJdt[i] = daIJdt(a[i], k[i], dadt[i], dkdt);
                  damdt[i] = damdt(daIJdt[i], da123dt[i], x);
                  d2adt2[i] = d2adt2(Tr[i], ac, a1, a2, a3, Tc);
                  d2a123dt2[i] = d2a123dt2(k123[i], a[i], dk123dt, dadt[i],
                                    d2adt2[i]);
                  d2aIJdt2[i] = d2aIJdt2(a[i], k[i], dadt[i], dkdt,
                                    d2adt2[i]);
                  d2amdt2[i] = d2amdt2(d2aIJdt2[i], d2a123dt2[i], x);

            }

// epanalhptikh diadikasia gia to prosdiorismo tou ogkou V apo piesh p
// kai thermokrasia T

            for (int j = 0; j < 10; j++) {

                  for (int i = 0; i < 2410; i++) {
                        V[j][i]= 1950.0;
                        double VOld;

                        do {
                              VOld = V[j][i];
                              V[j][i] = V1(T[i], p[j], am[i], bm, Rmol, VOld);

                        } while (Math.abs(VOld - V[j][i]) > 0.000001);
                  }
            }

// apothikeush stous antistoixous pinakes twn timwn twn megethwn pou
// eksartountai apo T kai p kalwmtas thn katallhlh sunarthsh me to
// katallhlo stoixeio eisodou

            for (int i = 0; i < 2410; i++) {
                  for (int j = 0; j < 10; j++) {
                        v[j][i]=v(V[j][i],Mol);
```

```
                        dpdt[j][i] = dpdt(damdt[i], bm, Rmol, V[j][i]);
                        dpdv[j][i] = dpdv(am[i], bm, Rmol, T[i], V[j][i]);

                        s[j][i] = s(damdt[i], bm, Rmol, R, Mol, T[i], To,
                                    V[j][i], so, po, A, B, C, D);
                        h[j][i] = h(damdt[i], am[i], bm, R, Mol, T[i], To,
                                    V[j][i], Uo, p[j], A, B, C, D);
                        cv[j][i] = cv (d2amdt2[i], bm, R, Mol, T[i], V[j][i],
                                    A, B, C, D);
                        cp[j][i] = cp (cv[j][i], Mol, T[i], dpdt[j][i],
                                    dpdv[j][i]);
                        kpv[j][i] = kpv (V[j][i], p[j], cv[j][i], cp[j][i],
                                    dpdv[j][i]);
                        ktv[j][i] = ktv (V[j][i], cv[j][i], dpdt[j][i], Mol);
                        kpt[j][i] = kpt (T[i], p[j], cv[j][i], cp[j][i],
                                    dpdt[j][i]);
                        g[j][i] = g (cp[j][i], cv[j][i]);
                        Z[j][i] = Z (p[j], V[j][i], T[i], Rmol);
                        atax[j][i] = atax (V[j][i], cv[j][i], cp[j][i],
                                    dpdv[j][i], Mol);

// mhdenismos timwn entos difasikhs perioxhs

                        if( v[j][i] < vsat[j]) {
                              v[j][i]=0;
                              s[j][i]=0;
                              h[j][i]=0;
                              cv[j][i]=0;
                              cp[j][i]=0;
                              kpv[j][i]=0;
                              ktv[j][i]=0;
                              kpt[j][i]=0;
                              g[j][i]=0;
                              Z[j][i]=0;
                              atax[j][i]=0;
                        }

                  }
            }

// apothikeush pinakwn ogkou v se arxeia Excel gia kathe piesh kai se olo to
// thermokrasiako euros

            ExcelFile filev= new ExcelFile();
            filev.addColumn("T", T);
            filev.addColumn("0.2 MPa", v[0]);
            filev.addColumn("0.3 MPa", v[1]);
            filev.addColumn("0.4 MPa", v[2]);
            filev.addColumn("0.5 MPa", v[3]);
            filev.addColumn("0.7 MPa", v[4]);
            filev.addColumn("1.0 MPa", v[5]);
            filev.addColumn("1.5 MPa", v[6]);
            filev.addColumn("2.0 MPa", v[7]);
            filev.addColumn("3.0 MPa", v[8]);
```

```
            filev.addColumn("4.0 MPa", v[9]);
            filev.export( "DiagrammaN v");

// apothikeush pinakwn eidikhs thermoxwrhtikothtas cv se arxeia Excel gia
// kathe piesh kai se olo to thermokrasiako euros

            ExcelFile filecv= new ExcelFile();
            filecv.addColumn("T", T);
            filecv.addColumn("0.2 MPa", cv[0]);
            filecv.addColumn("0.3 MPa", cv[1]);
            filecv.addColumn("0.4 MPa", cv[2]);
            filecv.addColumn("0.5 MPa", cv[3]);
            filecv.addColumn("0.7 MPa", cv[4]);
            filecv.addColumn("1.0 MPa", cv[5]);
            filecv.addColumn("1.5 MPa", cv[6]);
            filecv.addColumn("2.0 MPa", cv[7]);
            filecv.addColumn("3.0 MPa", cv[8]);
            filecv.addColumn("4.0 MPa", cv[9]);
            filecv.export( "DiagrammaN cv");

// apothikeush pinakwn eidikhs thermoxwrhtikothtas cp se arxeia Excel gia
// kathe piesh kai se olo to thermokrasiako euros

            ExcelFile filecp= new ExcelFile();
            filecp.addColumn("T", T);
            filecp.addColumn("0.2 MPa", cp[0]);
            filecp.addColumn("0.3 MPa", cp[1]);
            filecp.addColumn("0.4 MPa", cp[2]);
            filecp.addColumn("0.5 MPa", cp[3]);
            filecp.addColumn("0.7 MPa", cp[4]);
            filecp.addColumn("1.0 MPa", cp[5]);
            filecp.addColumn("1.5 MPa", cp[6]);
            filecp.addColumn("2.0 MPa", cp[7]);
            filecp.addColumn("3.0 MPa", cp[8]);
            filecp.addColumn("4.0 MPa", cp[9]);
            filecp.export( "DiagrammaN cp");

// apothikeush pinakwn isentropikou suntelesth kpv se arxeia Excel gia
// kathe piesh kai se olo to thermokrasiako euros

            ExcelFile filekpv= new ExcelFile();
            filekpv.addColumn("T", T);
            filekpv.addColumn("0.2 MPa", kpv[0]);
            filekpv.addColumn("0.3 MPa", kpv[1]);
            filekpv.addColumn("0.4 MPa", kpv[2]);
            filekpv.addColumn("0.5 MPa", kpv[3]);
            filekpv.addColumn("0.7 MPa", kpv[4]);
            filekpv.addColumn("1.0 MPa", kpv[5]);
            filekpv.addColumn("1.5 MPa", kpv[6]);
            filekpv.addColumn("2.0 MPa", kpv[7]);
            filekpv.addColumn("3.0 MPa", kpv[8]);
            filekpv.addColumn("4.0 MPa", kpv[9]);
            filekpv.export( "DiagrammaN kpv");
```

```
// apothikeush pinakwn isentropikou suntelesth ktv se arxeia Excel gia
// kathe piesh kai se olo to thermokrasiako euros

            ExcelFile filektv= new ExcelFile();
            filektv.addColumn("T", T);
            filektv.addColumn("0.2 MPa", ktv[0]);
            filektv.addColumn("0.3 MPa", ktv[1]);
            filektv.addColumn("0.4 MPa", ktv[2]);
            filektv.addColumn("0.5 MPa", ktv[3]);
            filektv.addColumn("0.7 MPa", ktv[4]);
            filektv.addColumn("1.0 MPa", ktv[5]);
            filektv.addColumn("1.5 MPa", ktv[6]);
            filektv.addColumn("2.0 MPa", ktv[7]);
            filektv.addColumn("3.0 MPa", ktv[8]);
            filektv.addColumn("4.0 MPa", ktv[9]);
            filektv.export( "DiagrammaN ktv");


// apothikeush pinakwn isentropikou suntelesth kpt se arxeia Excel gia
// kathe piesh kai se olo to thermokrasiako euros

            ExcelFile filekpt= new ExcelFile();
            filekpt.addColumn("T", T);
            filekpt.addColumn("0.2 MPa", kpt[0]);
            filekpt.addColumn("0.3 MPa", kpt[1]);
            filekpt.addColumn("0.4 MPa", kpt[2]);
            filekpt.addColumn("0.5 MPa", kpt[3]);
            filekpt.addColumn("0.7 MPa", kpt[4]);
            filekpt.addColumn("1.0 MPa", kpt[5]);
            filekpt.addColumn("1.5 MPa", kpt[6]);
            filekpt.addColumn("2.0 MPa", kpt[7]);
            filekpt.addColumn("3.0 MPa", kpt[8]);
            filekpt.addColumn("4.0 MPa", kpt[9]);
            filekpt.export( "DiagrammaN kpt");


// apothikeush pinakwn logou eidikwn thermoxwrhtikothtwn g se arxeia Excel
// gia kathe piesh kai se olo to thermokrasiako euros

            ExcelFile fileg= new ExcelFile();
            fileg.addColumn("T", T);
            fileg.addColumn("0.2 MPa", g[0]);
            fileg.addColumn("0.3 MPa", g[1]);
            fileg.addColumn("0.4 MPa", g[2]);
            fileg.addColumn("0.5 MPa", g[3]);
            fileg.addColumn("0.7 MPa", g[4]);
            fileg.addColumn("1.0 MPa", g[5]);
            fileg.addColumn("1.5 MPa", g[6]);
            fileg.addColumn("2.0 MPa", g[7]);
            fileg.addColumn("3.0 MPa", g[8]);
            fileg.addColumn("4.0 MPa", g[9]);
            fileg.export( "DiagrammaN gamma");
```

```
// apothikeush pinakwn taxuthtas hxou atax se arxeia Excel gia kathe piesh
// kai se olo to thermokrasiako euros

            ExcelFile fileatax= new ExcelFile();
            fileatax.addColumn("T", T);
            fileatax.addColumn("0.2 MPa", atax[0]);
            fileatax.addColumn("0.3 MPa", atax[1]);
            fileatax.addColumn("0.4 MPa", atax[2]);
            fileatax.addColumn("0.5 MPa", atax[3]);
            fileatax.addColumn("0.7 MPa", atax[4]);
            fileatax.addColumn("1.0 MPa", atax[5]);
            fileatax.addColumn("1.5 MPa", atax[6]);
            fileatax.addColumn("2.0 MPa", atax[7]);
            fileatax.addColumn("3.0 MPa", atax[8]);
            fileatax.addColumn("4.0 MPa", atax[9]);
            fileatax.export( "DiagrammaN atax");


// apothikeush pinakwn entropías s se arxeia Excel gia kathe piesh kai se
// olo to thermokrasiako euros

            ExcelFile files= new ExcelFile();
            files.addColumn("T", T);
            files.addColumn("0.2 MPa", s[0]);
            files.addColumn("0.3 MPa", s[1]);
            files.addColumn("0.4 MPa", s[2]);
            files.addColumn("0.5 MPa", s[3]);
            files.addColumn("0.7 MPa", s[4]);
            files.addColumn("1.0 MPa", s[5]);
            files.addColumn("1.5 MPa", s[6]);
            files.addColumn("2.0 MPa", s[7]);
            files.addColumn("3.0 MPa", s[8]);
            files.addColumn("4.0 MPa", s[9]);
            files.export( "DiagrammaN s");


// apothikeush pinakwn enthalpias h se arxeia Excel gia kathe piesh kai se
// olo to thermokrasiako euros

            ExcelFile fileh= new ExcelFile();
            fileh.addColumn("T", T);
            fileh.addColumn("0.2 MPa", h[0]);
            fileh.addColumn("0.3 MPa", h[1]);
            fileh.addColumn("0.4 MPa", h[2]);
            fileh.addColumn("0.5 MPa", h[3]);
            fileh.addColumn("0.7 MPa", h[4]);
            fileh.addColumn("1.0 MPa", h[5]);
            fileh.addColumn("1.5 MPa", h[6]);
            fileh.addColumn("2.0 MPa", h[7]);
            fileh.addColumn("3.0 MPa", h[8]);
            fileh.addColumn("4.0 MPa", h[9]);
            fileh.export( "DiagrammaN h");
```

```
// entolh tupwshs sthn othonh xrhsimopoumnehs mnhmhs RAM

      mem();

      }

// dhmiourgia entolhs tupwshs xrhsimopoioumenhs mnhmhs RAM

      public static void mem() {

            long max = Runtime.getRuntime().maxMemory() / 1000000;
            long total = Runtime.getRuntime().totalMemory() / 1000000;
            long used = total - Runtime.getRuntime().freeMemory() / 1000000;
            System.out.println(used + "/" + total + "  " + max);

      }

// dhmiourgia sunarthshs upologismou ogkou symfwna me thn katastatikh
// eksiswsh Peng-Robinson (cm3/mol)

      public static double V1 (double T, double p, double am, double bm,
                  double Rmol,double VOld) {

            double V = (1000 * Rmol * T) / (p+am/(Math.pow(VOld, 2)
                        + 2 * VOld * bm - (bm * bm))) + bm;

            return V;

      }

// dhmiourgia sunarthshs metratophs monadwn ogkou apo cm3/mol se m3/kg

      public static double v(double V, double Mol) {
            double v = V / (1000 * Mol);

            return v;
      }


// dhmiourgia sunarthshs upologismou anhgmenhs thermokrasias Tr gia kathe
// sustatiko

      public static double[] tr(double T, double[] Tc) {

            double[] Tr = new double[3];
            for (int j = 0; j < 3; j++) {
                  Tr[j] = T / Tc[j];
            }

            return Tr;

      }
```

```
// dhmiourgia sunarthshs upologismou statheras a (kPa cm6/mol2)
// stathera eksartwmnh apo th thermokrasia

	public static double[] a(double[] Tr, double[] ac, double[] a1,
			double[] a2, double[] a3) {

		double[] a = new double[3];
		for (int j = 0; j < 3; j++) {
			a[j] = 1.0 - Math.sqrt(Tr[j]);
			a[j] = 1.0 + a1[j] * a[j] + a2[j] * Math.pow(a[j], 2.0)
				+ a3[j] * Math.pow(a[j], 3.0);
			a[j] = ac[j] * Math.pow(a[j], 2.0);
		}

		return a;

	}

// dhmiourgia sunarthshs upologismou paragonta diortwshs kij
// paragwntas eksartwmenos apo th thermokrasia

	public static double[][] k(double T) {

		double[][] k = new double[3][3];

		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {

				if ((j == 0 && m == 1) || (j == 1 && m == 0)) {
					k[j][m] = 0.0;
				} else if ((j == 0 && m == 2) || (j == 2 && m == 0)){
					k[j][m] = 0.0;
				} else if ((j == 1 && m == 2) || (j == 2 && m == 1)){
					k[j][m] = 0.0;
				} else {
					k[j][m] = 0.0;
				}
			}
		}

		return k;

	}

// dhmiourgia sunarthshs upologismou paragonta diortwshs k123
// paragwntas eksartwmenos apo th thermokrasia

	public static double k123(double T) {

		double k123 = 0.0;

		return k123;

	}
```

```
// dhmiourgia sunarthshs upologismou statheras aij
// stathera eksartwmenh apo th thermokrasia


	public static double[][] aIJ(double[] a, double[][] k) {

		double[][] aIJ = new double[3][3];

		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {
				aIJ[j][m] = Math.sqrt(a[j] * a[m]) * (1.0 - k[j][m]);
			}
		}

		return aIJ;

	}



// dhmiourgia sunarthshs upologismou statheras a123
// stathera eksartwmenh apo th thermokrasia


	public static double a123(double k123, double[] a) {

		double a123 = k123 * Math.pow(a[0] * a[1] * a[2], (1 / 3.0));

		return a123;

	}



// dhmiourgia sunarthshs upologismou statheras am
// stathera eksartwmenh apo th thermokrasia


	public static double am(double[][] aIJ, double a123, double[] x) {

		double am = 0;
		for (int m = 0; m < 3; m++) {
			for (int j = 0; j < 3; j++) {
				am = x[j] * x[m] * aIJ[j][m] + am;
			}
		}
		am = am + 3.0 * a123 * x[0] * x[1] * x[2];

		return am;

	}
```

```
// dhmiourgia sunarthshs upologismou paragonta diortwshs lij

	public static double[][] l() {

		double[][] l = new double[3][3];
		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {
				if ((j == 0 & m == 1) || (j == 1 && m == 0)) {
					l[j][m] = 0.0;
				} else if ((j == 0 & m == 2) || (j == 2 && m == 0)) {
					l[j][m] = 0.0;
				} else if ((j == 1 & m == 2) || (j == 2 && m == 1)) {
					l[j][m] = 0.0;
				} else {
					l[j][m] = 0;
				}
			}
		}
		return l;

	}

// dhmiourgia sunarthshs upologismou statheras bij

	public static double[][] BIJ(double[] b, double[][] l) {

		double[][] BIJ = new double[3][3];
		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {
				BIJ[j][m] = (b[j] + b[m]) / 2.0;
				BIJ[j][m] = BIJ[j][m] * (1.0 - l[j][m]);
			}
		}

		return BIJ;

	}

// dhmiourgia sunarthshs upologismou statheras bm

	public static double bm(double[][] B, double[] x) {

		double bm = 0;
		for (int j = 0; j < 3; j++) {
			for (int m = 0; m < 3; m++) {
				bm = x[j] * x[m] * B[j][m] + bm;
			}
		}

		return bm;

	}
```

```
// dhmiourgia sunarthshs upologismou statheras dadt (paragwgos a)
// megethos eksartwmeno apo th thermokrasia

      public static double[] dadt(double[] Tr, double[] ac, double[] a1,
                  double[] a2, double[] a3, double[] Tc) {

            double[] dadt = new double[3];
            double[] B = new double[3];

            for (int j = 0; j < 3; j++) {
                  dadt[j] = 1.0 - Math.sqrt(Tr[j]);
                  dadt[j] = 1.0 + a1[j] * dadt[j]
                        + a2[j] * Math.pow(dadt[j], 2.0)
                        + a3[j] * Math.pow(dadt[j], 3.0);

                  B[j] = 1.0 - Math.sqrt(Tr[j]);
                  B[j] = a1[j] + 2.0 * a2[j] * B[j]
                        + 3.0 * a3[j]* Math.pow(B[j], 2.0);
                  B[j] = -(1.0 / (Math.sqrt(Tr[j]) * Tc[j])) * B[j];

                  dadt[j] = ac[j] * dadt[j] * B[j];

            }

            return dadt;

      }

// dhmiourgia sunarthshs upologismou paragonta diortwshs dkdt (paragwgos k)
// paragwntas eksartwmenos apo th thermokrasia

      public static double[][] dkdt() {

            double[][] dkdt = new double[3][3];

            for (int j = 0; j < 3; j++) {
                  for (int m = 0; m < 3; m++) {
                        if ((j == 0 && m == 1) || (j == 1 && m == 0)) {
                              dkdt[j][m] = 0.0;
                        } else if ((j == 0 && m == 2) || (j == 2 && m == 0)){
                              dkdt[j][m] = 0.0;
                        } else if ((j == 1 && m == 2) || (j == 2 && m == 1)){
                              dkdt[j][m] = 0.0;
                        } else {
                              dkdt[j][m] = 0.0;
                        }
                  }
            }

            return dkdt;

      }
```

```
// dhmiourgia sunarthshs upologismou statheras da123dt (paragwgos a123)
// megethos eksartwmeno apo th thermokrasia

      public static double da123dt(double k123, double[] a, double dk123dt,
                  double[] dadt) {

            double B = dadt[0] * a[1] * a[2] + dadt[1] * a[0] * a[2]
                        + dadt[2]* a[1] * a[0];
            B = B / (Math.pow((a[0] * a[1] * a[2]), (2.0 / 3.0)));

            double da123dt = Math.pow((a[0] * a[1] * a[2]), (1.0 / 3.0));
            da123dt = da123dt * dk123dt + k123 * B;

            return da123dt;

      }

// dhmiourgia sunarthshs upologismou statheras daijdt (paragwgos aij)
// megethos eksartwmeno apo th thermokrasia

      public static double[][] daIJdt(double[] a, double[][] k,
                  double[] dadt,double[][] dkdt) {

            double[][] daIJdt = new double[3][3];

            for (int j = 0; j < 3; j++) {
                  for (int m = 0; m < 3; m++) {
                        daIJdt[j][m] = 0.5 * Math.sqrt(a[m] / a[j]) * dadt[j]
                                    * (1.0 - k[j][m]);
                        daIJdt[j][m] = 0.5 * Math.sqrt(a[j] / a[m]) * dadt[m]
                                    * (1.0 - k[j][m]) + daIJdt[j][m];
                        daIJdt[j][m] = daIJdt[j][m] - Math.sqrt(a[j] * a[m])
                                    * dkdt[j][m];
                  }
            }

            return daIJdt;
      }
// dhmiourgia sunarthshs upologismou statheras damdt (paragwgos am)
// megethos eksartwmeno apo th thermokrasia

      public static double damdt(double[][] daIJdt, double da123dt,
                  double[] x) {

            double damdt = 0;
            for (int m = 0; m < 3; m++) {
                  for (int j = 0; j < 3; j++) {
                        damdt = x[j] * x[m] * daIJdt[j][m] + damdt;
                  }
            }
            damdt = damdt + 3.0 * da123dt * x[0] * x[1] * x[2];

            return damdt;
      }
```

```
// dhmiourgia sunarthshs upologismou paragwgou pieshs ws pros thermokrasia
// megethos eksartwmeno apo th thermokrasia kai th piesh (kpa/K)

      public static double dpdt(double damdt, double bm, double Rmol,
                  double V) {

            double dpdt = 1000 * Rmol / (V - bm)
                        - (damdt / (Math.pow(V, 2.0)
                        + 2 * V * bm - Math.pow(bm, 2.0)));

            return dpdt;

      }

// dhmiourgia sunarthshs upologismou paragwgou pieshs ws pros ogko
// megethos eksartwmeno apo th thermokrasia kai th piesh (kPa/(cm3/mol))

public static double dpdv(double am, double bm, double Rmol, double T,
                  double V) {

            double dpdv = Math.pow(V, 2) + 2 * V * bm - Math.pow(bm, 2);
            dpdv = (-1000 * Rmol * T / (Math.pow((V - bm), 2.0)))
                        + ((am * 2.0 * (V + bm)) / (Math.pow(dpdv, 2.0)));

            return dpdv;

      }

// dhmiourgia sunarthshs upologismou paragwgou entropias s (kJ/kgK)
// megethos eksartwmeno apo th thermokrasia kai th piesh

      public static double s(double damdt, double bm, double Rmol, double R,
                  double Mol, double T, double To, double V, double so,
                  double po,double A, double B, double C, double D) {

            double r = 1 / V;

            double s = Math.log(Math.abs((-bm * r + 1.0 - Math.sqrt(2.0))
                        / (-bm * r + 1.0 + Math.sqrt(2.0))));
            s = s - Math.log(Math.abs((1.0 - Math.sqrt(2.0))
                        / (1.0 + Math.sqrt(2.0))));
            s = damdt * s / (bm * Math.sqrt(8.0));
            s = Rmol * Math.log(1.0 - r * bm) + s;
            s = s / (1000 * Mol);
            s = -R * Math.log(r * 1000 * Rmol * T / po) + s;
            s = -A * Math.log(To) - B * To - 0.5 * C * Math.pow(To, 2.0)
                        - (1/3.0) * D * Math.pow(To, 3.0) + s;
            s = A * Math.log(T) + B * T + 0.5 * C * Math.pow(T, 2.0)
                        + (1/3.0) * D * Math.pow(T, 3.0) + so + s;

            return s;

      }
```

```
// dhmiourgia sunarthshs upologismou paragwgou enthalpias h (kJ/kg)
// megethos eksartwmeno apo th thermokrasia kai th piesh

      public static double h(double damdt, double am, double bm, double R,
                  double Mol, double T, double To, double V, double Uo, double
                  p, double A, double B, double C, double D) {

            double r = 1 / V;

            double h = Math.log(Math.abs((-bm * r + 1.0 - Math.sqrt(2.0))
                        / (-bm * r + 1.0 + Math.sqrt(2.0))));
            h = h - Math.log(Math.abs((1.0 - Math.sqrt(2.0))
                        / (1.0 + Math.sqrt(2.0))));
            h = (-am + T * damdt) * h / ((bm * Math.sqrt(8.0)));
            h = p / r + h;
            h = h / (1000 * Mol);
            h = -R * (T - To) + h;
            h = -A * To - 0.5 * B * Math.pow(To, 2.0) - (1 / 3.0) * C
                  * Math.pow(To, 3.0) - 0.25 * D * Math.pow(To, 4.0) +h;
            h = A * T + 0.5 * B * Math.pow(T, 2.0) + (1 / 3.0) * C
                  * Math.pow(T, 3.0) + 0.25 * D * Math.pow(T, 4.0) + h;
            h = Uo + h;

            return h;

      }

// dhmiourgia sunarthshs upologismou ths 2hs paragwgou ths statheras a
// megethos eksartwmeno apo th thermokrasia

      public static double[] d2adt2(double[] Tr, double[] ac, double[] a1,
                  double[] a2, double[] a3, double[] Tc) {

            double[] d2adt2 = new double[3];
            double[] B = new double[3];
            double[] C = new double[3];
            double[] D = new double[3];
            for (int j = 0; j < 3; j++) {
                  d2adt2[j] = 1.0 - Math.sqrt(Tr[j]);
                  d2adt2[j] = 1.0 + a1[j] * d2adt2[j]
                        + a2[j] * Math.pow(d2adt2[j], 2.0)
                        + a3[j] * Math.pow(d2adt2[j], 3.0);
                  B[j] = 1.0 - Math.sqrt(Tr[j]);
                  B[j] = a1[j] + 2.0 * a2[j] * B[j]
                        + 3.0 * a3[j] * Math.pow(B[j], 2.0);
                  C[j] = ac[j] / (2.0 * Tr[j] * Math.pow(Tc[j], 2.0));
                  D[j] = 1.0 - Math.sqrt(Tr[j]);
                  D[j] = 6.0 * a3[j] * D[j] + 2.0 * a2[j];
                  d2adt2[j] = C[j]* (d2adt2[j] * B[j] / (Math.sqrt(Tr[j]))
                        + Math.pow(B[j], 2.0) + d2adt2[j] * D[j]);
            }

            return d2adt2;
      }
```

```
// dhmiourgia sunarthshs upologismou ths 2hs paragwgou ths statheras a123
// megethos eksartwmeno apo th thermokrasia

      public static double d2a123dt2(double k123, double[] a, double dk123dt,
                  double[] dadt, double[] d2adt2) {

            double A = Math.pow((a[0] * a[1] * a[2]), (1.0 / 3.0));
            double B = dadt[0] * a[1] * a[2] + dadt[1] * a[0] * a[2]
                  + dadt[2]* a[1] * a[0];
            double C = B / (3.0 * Math.pow(A, 2.0));
            double D = d2adt2[0] * a[1] * a[2]
                  + d2adt2[1] * a[0] * a[2] + d2adt2[2] * a[1] * a[0]
                  + 2.0 * (dadt[0] * dadt[1] * a[2]
                  + dadt[2] * dadt[1] * a[0] + dadt[0] * dadt[2] * a[1]);
            D = D / (3.0 * Math.pow(A, 2.0));
            D = -(2.0 / 9.0) * Math.pow(B, 2.0) / Math.pow(A, 5.0) + D;

            double d2a123dt2 = 2.0 * C * dk123dt + k123 * D;

            return d2a123dt2;

      }

// dhmiourgia sunarthshs upologismou ths 2hs paragwgou ths statheras aij
// megethos eksartwmeno apo th thermokrasia

      public static double[][] d2aIJdt2(double[] a, double[][] k,
                  double[] dadt, double[][] dkdt, double[] d2adt2) {

            double[][] d2aIJdt2 = new double[3][3];

            for (int j = 0; j < 3; j++) {
                  for (int m = 0; m < 3; m++) {
                        d2aIJdt2[j][m] = -dkdt[j][m] * (a[j] * dadt[m]
                              + a[m] * dadt[j]) / Math.sqrt(a[j] * a[m]);

                        d2aIJdt2[j][m] = d2aIJdt2[j][m] - (1 - k[j][m])
                              * Math.pow(a[j] * dadt[m]
                              + a[m] * dadt[j], 2.0)
                              / (4 * Math.pow(Math.sqrt(a[j] * a[m]), 3.0));

                        d2aIJdt2[j][m] = d2aIJdt2[j][m] + (1 - k[j][m])
                              * (a[j] * d2adt2[m] + a[m] * d2adt2[j]
                              + 2.0 * dadt[j] * dadt[m])
                              / (2 * (Math.sqrt(a[j] * a[m])));
                  }
            }

            return d2aIJdt2;

      }
```

```
// dhmiourgia sunarthshs upologismou ths 2hs paragwgou ths statheras am
// megethos eksartwmeno apo th thermokrasia

      public static double d2amdt2(double[][] d2aIJdt2, double d2a123dt2,
                  double[] x) {

            double d2amdt2 = 0;
            for (int m = 0; m < 3; m++) {
                  for (int j = 0; j < 3; j++) {
                        d2amdt2 = x[j] * x[m] * d2aIJdt2[j][m] + d2amdt2;
                  }
            }
            d2amdt2 = d2amdt2 + 3.0 * d2a123dt2 * x[0] * x[1] * x[2];

            return d2amdt2;

      }

// dhmiourgia sunarthshs upologismou paragwgou eidikhs thermoxwrhtikothtas
// upo stathero ogko cv (kJ/kg K)
// megethos eksartwmeno apo th thermokrasia kai th piesh

      public static double cv(double d2amdt2, double bm, double R,
                  double Mol, double T, double V, double A, double B,
                  double C, double D) {

            double r = 1 / V;
            double cv = Math.log(Math.abs((-bm * r + 1.0 - Math.sqrt(2.0))
                        / (-bm * r + 1.0 + Math.sqrt(2.0))));
            cv = cv - Math.log(Math.abs((1.0 - Math.sqrt(2.0))
                        / (1.0 + Math.sqrt(2.0))));
            cv = T * d2amdt2 * cv / (1000 * Mol * bm * Math.sqrt(8.0));
            cv = cv - R;
            cv = cv + A + B * T + C * Math.pow(T, 2.0)
                        + D * Math.pow(T, 3.0);

            return cv;

      }

// dhmiourgia sunarthshs upologismou paragwgou eidikhs thermoxwrhtikothtas
// upo statherh piesh cp (kJ/kg K)
// megethos eksartwmeno apo th thermokrasia kai th piesh

      public static double cp(double cv, double Mol, double T, double dpdt,
                  double dpdv) {

            double cp = cv - T * Math.pow(dpdt, 2.0) / (dpdv * 1000 * Mol);

            return cp;

      }
```

```
// dhmiourgia sunarthshs upologismou isentropikou suntelesth kpv
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double kpv(double V, double p, double cv, double cp,
                  double dpdv) {

            double kpv = -V * cp * dpdv / (p * cv);

            return kpv;

      }

// dhmiourgia sunarthshs upologismou isentropikou suntelesth ktv
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double ktv(double V, double cv, double dpdt, double Mol){

            double ktv = 1.0 + V * dpdt / (cv * 1000.0 * Mol);

            return ktv;

      }

// dhmiourgia sunarthshs upologismou isentropikou suntelesth kpt
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double kpt(double T, double p, double cv, double cp,
                  double dpdt) {

            double kpt = T * dpdt / (T * dpdt + p * (cv / cp - 1.0));

            return kpt;

      }

// dhmiourgia sunarthshs upologismou logou eidikwn thermoxwrhtikothtwn g
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double g(double cv, double cp) {

            double g = cp / cv;

            return g;
      }

// dhmiourgia sunarthshs upologismou suntelesth sumpiestothtas Z
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double Z(double p, double V, double T, double Rmol) {

            double Z = p * V / (1000 * Rmol * T);

            return Z;
      }
```

```
// dhmiourgia sunarthshs upologismou taxuthtas hxou a (m/s)
// megethos eksartwmeno apo th thermokrasia kai ton ogko

      public static double atax(double V, double cv, double cp, double dpdv,
                  double Mol) {

            double atax = -Math.pow(V, 2.0) * cp * dpdv / (cv * Mol);
            atax = Math.sqrt(atax);

            return atax;

      }
```

Βιβλιογραφία

1. Jiangtao Wu, Yingjie Chu, Jing Hu, Zhigang Liu : Performance of mixture refrigerant R152a / R125 / R32 in domestic air-conditioner. International Journal of Refrigeration 32, 2009

2. NIST standard reference database 23, version 8.0, 2007

3. Stegou-Sagia A., X.Kakatsios, Damanakis M. : Ozone friendly binary blends R32/R134a and the ternary R407b. International Journal of Energy 29, 2004

4. Ιστοσελίδα της United States Environmental Protection Agency – EPA, http://www.epa.gov/Ozone/geninfo/gwps.html

5. Reid R.C., Prausnitz J.M., Poling B.E. : The properties of gases and liquids, 4th ed. Singapore: McGraw-Hill Book Company, 1988

6. Perry R.H.,Green D.W. : Perry's chemical engineer's handbook 6th ed. Singapore: McGraw-Hill Book Company, 1984

7. Stegou-Sagia A., Damanakis M. : Binary and ternary blends of R134a as alternative refrigerants to R22. International Journal of Energy Conversion and Management 41, 2000

8. ICI chemicals and polymers, ICI KLEA 407b engineer's tables. Runcorn, Cheshire England, 1994

9. Smith J.M., Van Ness H.C., Abbott M.M : Introduction to chemical engineering thermodynamics. 5th ed. Singapore: McGraw-Hill Book Company, 1996

10. Mc Linden Marc O. : Thermodynamic properties of CFC alternatives: A survey of the available data. International Journal of Refrigeration 13, 1990