



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Barcelona Est

TREBALL FI DE GRAU

Grau en Enginyeria Mecànica

**MODELITZACIÓ DE GASOS EN CONDICIONS
SUPERCRÍTIQUES. VALIDACIÓ DEL CODI RHEA.**



Memòria i Annexos

Autor: Josep Gorres Salvadó
Director: Joan Grau Barceló
Convocatòria: Juny 2021

Resum

El codi de simulació *flowsolverRHEA*, és un codi de simulació de gasos en condicions supercrítiques i està dissenyat per simulacions a gran escala. El present projecte s'integra en la mecànica de fluids i es basa en la validació del codi de simulació *flowsolverRHEA* utilitzant l'equació d'estat de Peng-Robinson.

Es comença definint el codi de simulació, l'equació de Peng-Robinson i els 7 coeficients polinomials de Janaf-NASA. Un cop explicat el codi es procedeix a explicar el funcionament del codi i totes les seves limitacions.

Per comprovar el funcionament de l'equació d'estat de Peng-Robinson es compara els resultats del codi amb les dades publicades del NIST i l'equació d'estat de gas ideal. En els annexos hi ha gràfiques que s'obtenen de les simulacions per diferents substàncies, pressions i temperatures. Aquestes gràfiques serveixen d'ajut per valorar si el *flowsolverRHEA* obtindrà un bon resultat abans d'utilitzar-lo. En la majoria de casos s'ha simulat fins a gairebé 30 vegades la pressió crítica.

Un cop validat el codi de simulació es presenten els resultats obtinguts de les simulacions pels diferents gasos en forma de gràfiques i taules de resultats. S'ha de tenir en compte que les dades del NIST tenen certa incertesa. No tenir un resultat exacte a les dades del NIST no significa que el *flowsolverRHEA* no proporcioni un bon resultat. Quan el *flowsolverRHEA* es compara amb el model de gas ideal s'observa que és molt més fiable i exacte.

Resumen

El código de simulación flowsolverRHEA, es un código de simulación de gases en condiciones supercríticas y está diseñado para simulaciones a gran escala. El presente proyecto se integra en la mecánica de fluidos y se basa en la validación del código de simulación flowsolverRHEA utilizando la ecuación de estado de Peng-Robinson.

Se empieza definiendo el código de simulación, la ecuación de Peng-Robinson y los 7 coeficientes polinomiales de Janaf-NASA. Una vez explicado el código se procede a explicar el funcionamiento del código y todas sus funciones.

Para comprobar el funcionamiento de la ecuación de estado de Peng-Robinson se comparan los resultados del código con las dadas publicadas en el NIST y la ecuación de estados de gas ideal. En los anexos hay gráficas que se obtienen de las simulaciones para diferentes sustancias, presiones y temperaturas. Estas gráficas sirven de ayuda para valorar si el flowsolverRHEA tendrá un buen resultado antes de utilizarlo. En la mayoría de casos se ha simulado hasta casi 30 veces la presión crítica.

Una vez validado el código de simulación se presentan los resultados obtenidos de las simulaciones para diferentes gases en forma de gráficas y de tablas. Hay que tener en cuenta que las dadas del NIST tienen cierta incerteza. No tener un resultado exacto a las dadas del NIST no significa que el flowsolverRHEA no de un buen resultado. Cuando el flowsolverRHEA se compara con el modelo de gas ideal se observa que es mucho mas fiable y exacto.

Abstract

The *flowsolverRHEA* simulation code, is a gas simulation code in supercritical conditions and is designed for large-scale simulations. The present project is integrated into fluid mechanics and is based on the validation of the *flowsolverRHEA* simulation code using the Peng-Robinson state equation.

It begins by defining the simulation code, the Peng-Robinson equation, and the 7 Janaf-NASA polynomial coefficients. Once the code has been explained, the operation of the code and all its limitations are explained.

To test the operation of the Peng-Robinson state equation, the results of the code are compared with the published data from NIST and the ideal gas state equation. In the appendices there are graphs obtained from the simulations for different substances, pressures and temperatures. These graphs help to assess whether the *flowsolverRHEA* will perform well before use. In most cases, the critical pressure has been simulated up to almost 30 times.

Once the simulation code has been validated, the results obtained from the simulations for the different gases are presented in the form of graphs and tables of results. It should be noted that the NIST data has some uncertainty. Not having an exact result in NIST data does not mean that *flowsolverRHEA* does not provide a good result. When the *flowsolverRHEA* is compared to the ideal gas model it is observed to be much more reliable and accurate.

Agraïments

En primer lloc vull agrair al tutor del TFG, en Joan Grau, per haver-me donat l'oportunitat d'executar aquest projecte. Tot i les dificultats de dur a terme un projecte telemàtic a causa de la pandèmia, voldria agrair tota l'ajuda que m'ha donat el tutor durant la realització d'aquest. Gràcies a ell he pogut adquirir més coneixements sobre mecànica de fluids, termodinàmica i programació.

També voldria agrair a la meva família, que tots aquests anys que he estat realitzant el grau sempre han estat al meu costat donant-me suport i per haver recolzat sempre les meves decisions.

Índex

RESUM	I
RESUMEN	II
ABSTRACT	III
AGRAÏMENTS	IV
1. PREFACI	1
1.2. Origen del treball	1
1.3. Motivació	1
1.4. Requeriments previs	1
2. INTRODUCCIÓ	3
2.1. Objectius del treball	3
2.2. Abast del treball	3
3. CONCEPTES BÀSICS	5
3.1. Diagrames de fase	5
3.1.1. Punt crític	7
3.2. Equacions d'estat	8
3.3. NIST	9
4. FLOWSOLVERRHEA	11
4.1. Necessitat de crear un solver en flux supercrític	11
4.2. FlowsolverRHEA	12
4.3. Equació d'estat de Peng-Robinson	12
4.3.1. Exemple de càlcul amb l'equació d'estat Peng-Robinson	15
4.3.2. Limitacions de l'equació	16
4.4. Janaf-NASA 7-coeficients polinomials	17
5. PROCEDIMENT	19
5.1. Crear llibreria de substàncies	19
5.2. Propietats NIST	21
5.3. Postprocessat	24
5.4. Limitació Peng-Robinson	27
5.5. Intervals de simulació	30
5.6. Comparació Peng-Robinson amb Gas Ideal	31

6. FUNCIONAMENT	32
6.1. Configuració fitxers.....	32
6.1.1. Configuration_file.yaml	32
6.1.2. MyRHEA.cpp	33
6.1.3. post_process_plot_script.py.....	33
6.2. Execució fitxers	34
7. RESULTATS	35
7.1. Nitrogen	35
7.1.1. 6,79 MPa	35
7.1.2. 16,97 Mpa	36
7.1.3. 33,95 MPa	38
7.1.4. 60 MPa	39
7.1.5. 100 MPa	41
7.2. Taules resultats.....	43
ANÀLISI DE IMPACTE AMBIENTAL DEL PROJECTE	45
ANÀLISI ECONÒMIC DEL PROJECTE	46
CONCLUSIONS	47
BIBLIOGRAFIA	48
ANNEX A: DOCUMENTS	49
A1. Llibreria de substàncies	49
A2. Substances_selection	58
A3. Postprocessat original	59
A4. Postprocessat NIST, RHEA	62
A5. Postprocessat NIST, RHEA i IDEAL GAS	66
A6. Programa Scilab	70
ANNEX B: GRÀFIQUES SCILAB	71
Diòxid de carboni (14,75 MPa).....	71
Diòxid de carboni (36,89 MPa).....	71
Diòxid de carboni (73,77 MPa).....	72
Diòxid de carboni (140 MPa).....	72
Diòxid de carboni (200 MPa).....	73
Dodecà (3,63 MPa)	73

Dodecà (9,09 MPa)	74
Dodecà (18,17 MPa)	74
Dodecà (50 MPa)	75
Hidrogen (2,59 MPa).....	75
Hidrogen (6,48 MPa).....	76
Hidrogen (12,96 MPa).....	76
Hidrogen (20 MPa).....	77
Hidrogen (30 MPa).....	77
Metà (9,19 MPa)	78
Metà (22,99 MPa)	78
Metà (45,99 MPa)	79
Metà (80 MPa)	79
Metà (100 MPa)	80
Oxigen (10,09 MPa)	80
Oxigen (25,22 MPa)	81
Oxigen (50,43 MPa)	81
Oxigen (65 MPa)	82
Oxigen (82 MPa)	82
Nitrogen (6,79 MPa)	83
Nitrogen (16,97 MPa)	83
Nitrogen (33,95 MPa)	84
Nitrogen (60 MPa)	84
Nitrogen (100 MPa)	85
ANNEX C: GRÀFIQUES RHEA	86
Diòxid de carboni (14,75 MPa)	86
Diòxid de carboni (36,89 MPa)	87
Diòxid de carboni (73,77 MPa)	89
Diòxid de carboni (140 MPa)	90
Diòxid de carboni (200 MPa)	92
Dodecà 3,63 MPa	93
Dodecà 9,09 MPa	95
Dodecà 18,17 MPa	96
Dodecà 50 MPa	98
Hidrogen (2,59 MPa).....	99

Hidrogen (6,48 MPa)	101
Hidrogen (12,96 MPa)	102
Hidrogen (20 MPa)	104
Hidrogen (30 MPa)	105
Metà (9,19 MPa).....	107
Metà (22,99 MPa).....	108
Metà (45,99 MPa).....	110
Metà (80 MPa).....	111
Metà (100 MPa).....	113
Oxigen (10,09 MPa)	114
Oxigen (25,22 MPa)	116
Oxigen (50,43 MPa)	117
Oxigen (65 MPa)	119
Oxigen (82 MPa)	120

1. Prefaci

1.2. Origen del treball

Aquest treball ve donat per diversos factors. En un principi, volia buscar una empresa on realitzar les pràctiques i allí elaborar el TFG, però donades les dificultats actuals per trobar feina, no va ser possible. Tot seguit, per la COVID-19, la qual va causar el confinament i es van suprimir les classes presencials, no tenia contacte amb el professorat, per tant, va ser molt més difícil contactar amb ells per buscar una proposta de TFG. Després de contactar amb diferents professors, molts no em donaven resposta o no podien portar un TFG per la situació actual. Finalment, el Joan Grau em va proposar un projecte relacionat amb el desenvolupament d'un nou programa de simulació de fluids i em va semblar molt interessant l'objectiu del projecte, fet que em va ajudar a acabar-me de decidir sobre què tractaria la present recerca.

1.3. Motivació

Com a estudiant d'enginyeria volia aplicar els coneixements adquirits durant el grau i aprofundir-los més.

La principal motivació va ser realitzar un treball relacionat amb la mecànica de fluids o la termodinàmica, i aquest treball engloba els dos camps. També volia que fossa un tema innovador, com és aquest cas, i d'utilitat per una posterior aplicació a la pràctica. Un dels meus principals objectius és que el meu treball aportat pugui servir pel desenvolupament del flowsolverRHEA i per la validació d'aquest.

1.4. Requeriments previs

Des d'un inici no m'esperava haver de programar, però donada la constitució del treball he hagut de programar petits codis amb Python i Scilab. Ja tenia coneixements de programació amb aquestes eines, però he hagut d'aprofundir un poc més per realitzar el projecte correctament. Realment, el fet de programar m'ha ajudat a agilitzar algunes tasques del projecte.

Els flowsolverRHEA funciona amb el sistema operatiu Linux. No havia utilitzat mai aquest sistema operatiu anteriorment, però he après a utilitzar-lo amb agilitat.

Finalment, ja tenia coneixements previs de termodinàmica i mecànica de fluids, però he hagut d'aprofundir més en alguns aspectes per entendre el funcionament del flowsolverRHEA.

2. Introducció

La dinàmica de fluids computacional (CFD) és una de les rames de la mecànica de fluids que utilitza els mètodes numèrics i algoritmes per resoldre problemes sobre flux de fluids. Els ordenadors són utilitzats per simular la interacció entre líquids i gasos en superfícies complexes per a l'enginyeria. Tot i això, els resultats que obtenen són aproximats i tenen un marge d'error. La continua investigació permet incorporar software que augmenti la velocitat de càlcul, en disminueixi l'error i permeti utilitzar superfícies més complexes.

Des de l'any passat, el Departament de Mecànica de fluids de la Universitat Politècnica de Catalunya ha estat desenvolupant un nou programa CFD per simular fluids a pressions supercrítiques. En aquest projecte s'estudiarà i es verificarà el seu funcionament.

2.1. Objectius del treball

L'objectiu d'aquest treball és validar el flowsolverRHEA comparant-lo amb les dades del NIST, amb diferents substàncies, pressions i rangs de temperatures. Els objectius es dividiran en dos parts: teòrica i pràctics.

- Objectius teòrics: entendre el funcionament del flowsolverRHEA en la modelització de gasos. Estudiar el comportament de l'equació d'estat de Peng-Robinson.
- Objectius pràctics: realitzar simulacions amb el flowsolverRHEA i posar-lo a prova. Començar en simulacions bàsiques per entendre el seu funcionament i anar progressant per establir els seus rangs de treball.

2.2. Abast del treball

Al tractar-se d'un treball telemàtic no ha fet falta desplaçar-se al campus per realitzar les simulacions, cosa que en la situació actual m'ha comportat una gran avantatge. El projecte intentarà realitzar el màxim de simulacions possible amb el flowsolverRHEA per tenir un nombre major de gràfiques estudiant el major nombre de substàncies i veure així el seu funcionament.

3. Conceptes bàsics

3.1. Diagrames de fase

L'estudi de la termodinàmica involucra el comportament de les substàncies pures, que son aquelles que posseeixen una composició definida i son químicament homogènies, no importa si estan en més d'una fase sempre i quan la seva composició química sigui constant. Definir les transicions entre diferents fases d'una substància és de vital importància tecnològica.

Per entendre el canvi de fase d'una substància imagineu un bloc d'aigua en estat sòlid al qual se li aplica calor. La temperatura d'aquest bloc augmenta de forma proporcional a la calor que se li aporta fins que comença a fondre's. La temperatura del sòlid es manté constant mentre es produeix el canvi de fase. Un cop ha passat a estat líquid la temperatura de l'aigua augmenta proporcionalment a la calor aportada. L'augment de temperatura per unitat d'energia aportada serà diferent perquè ha canviat de fase. Quan l'aigua començarà a evaporar-se, la temperatura es mantindrà constant fins que s'haurà evaporat per complet. Després seguirà augmentant la seva temperatura. Aquesta seqüència s'il·lustra a la Figura 3-1 que representa el volum específic en funció de la temperatura.

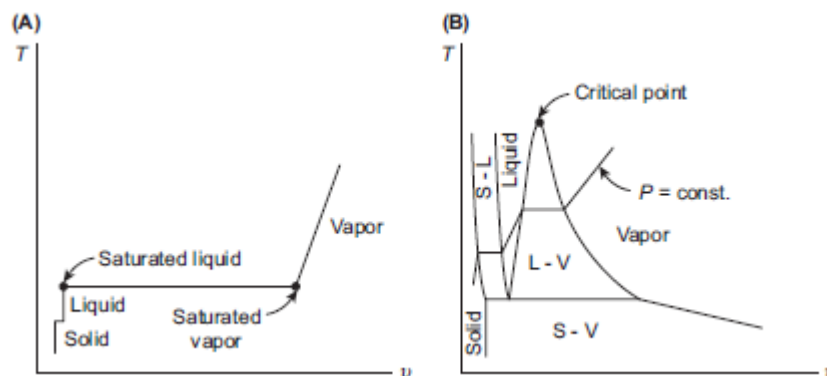


Figura 3-1. Diagrames de fase (Font: Chapter 2 – Properties of Pure Substances, Bahman Zohuri)

També es pot expressar en funció de la pressió:

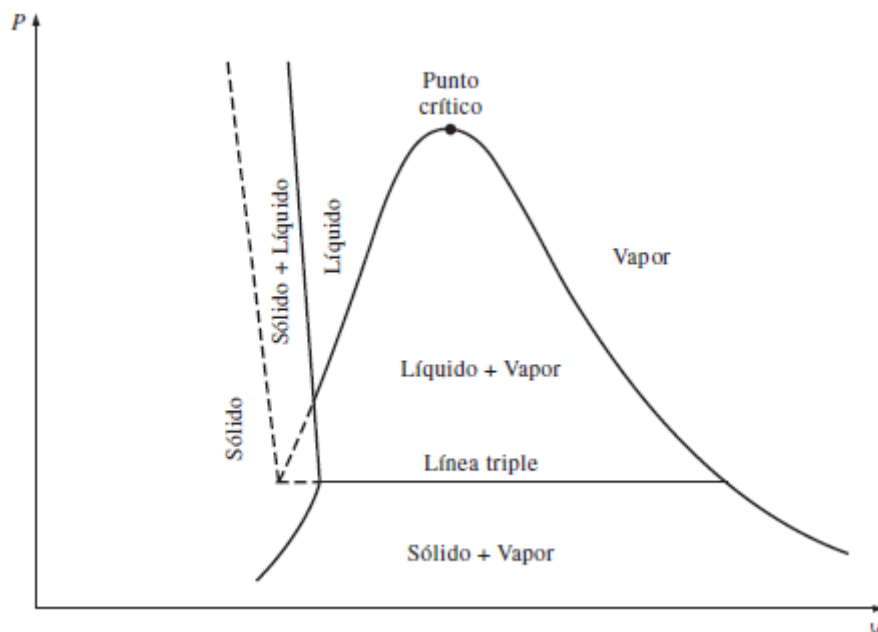


Figura 3-2. Diagrama de fase (Font: Termodinàmica – Cengel 7a edició)

Si es representa en funció de la temperatura i la pressió, passa a ser un diagrama de tres dimensions. La Figura 3-3 mostra el comportament típic de les substàncies.

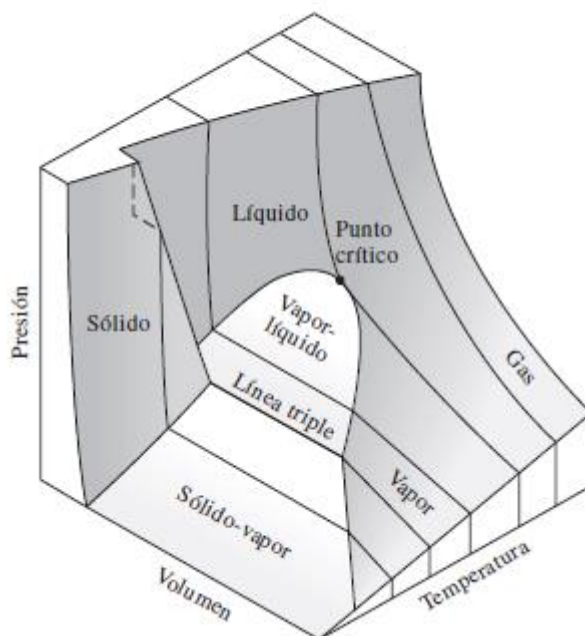


Figura 3-3. Diagrama de fases en tres dimensions (Font: Termodinàmica – Cengel 7a edició)

3.1.1. Punt crític

El punt crític es defineix com el punt en què els estats de líquid saturat i vapor saturat son idèntics. Una altra definició és que quan la densitat del líquid i del vapor son iguals. Per sobre del punt crític, ja sigui pressió o temperatura, el fluid es troba en estat supercrític. No pot identificar-se com un líquid o vapor, però posseeix propietats entre les d'un gas i d'un líquid. Es pot apreciar una transició entre unes propietats més properes a líquid i unes propietats més properes a gas. A la Figura 3-4 s'observa el punt crític el CO₂ i la zona de fluid supercrític.

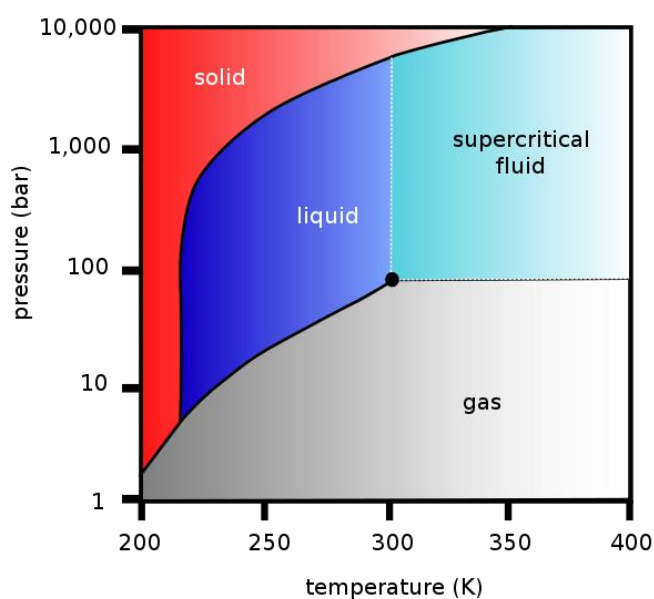


Figura 3-4. Representació punt crític en diagrama de fases (Font: <https://www.pngwing.com/es/free-png-yygsr>)

A continuació a la figura 3-5 es mostra la transició del Nitrogen d'estat líquid a supercrític i el canvi de la densitat.

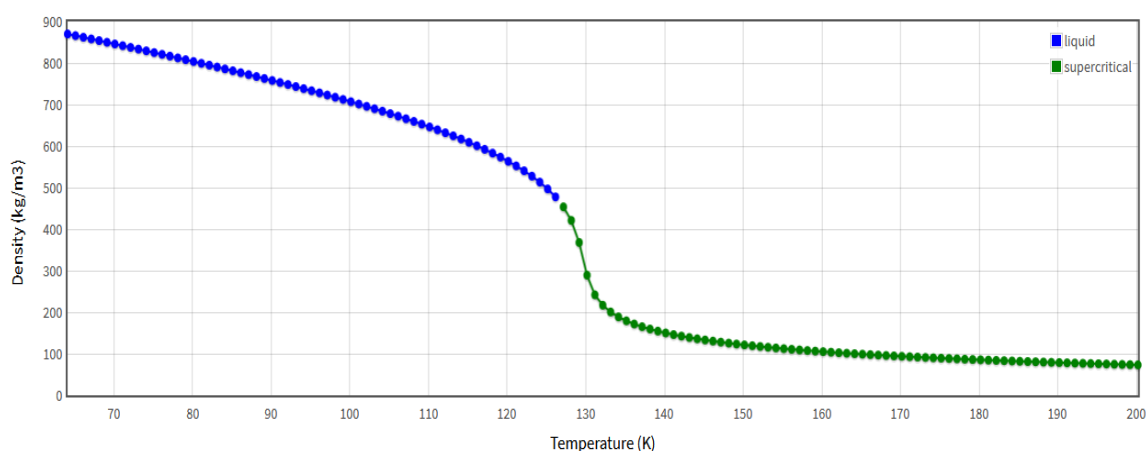


Figura 3-5. Representació de la densitat del Nitrogen per una pressió de 4MPa (Font: Webbook NIST)

3.2. Equacions d'estat

Hi ha propietats intensives com la pressió i la temperatura, i hi ha propietats extensives com l'entalpia, l'energia interna, l'entropia i el volum. Qualsevol equació que relacioni la temperatura, pressió i volum específic d'una substància s'anomena equació d'estat.

Imagineu un sistema tancat en un recipient, on la quantitat de matèria es manté fixada, i es poden mesurar fàcilment la pressió, volum i temperatura. Si pel volum s'estableix un valor arbitrari i la temperatura es manté a un valor específic, la pressió es fixarà en un valor definit. Un cop que el volum i la temperatura s'escullen, llavors els valor de pressió es fixa en equilibri. Existeix una equació d'equilibri, anomenada equació d'estat, que relaciona les tres coordenades termodinàmiques. Qualsevol sistema termodinàmic té la seva pròpia equació d'estat, des d'una simple relació matemàtica fins a alguns casos on no es poden expressar en termes de funcions matemàtiques simples.

Hi ha diverses equacions d'estat, algunes de senzilles i d'altres molt més complexes. La més senzilla i coneguda per substàncies en fase gasosa, és l'equació d'estat de gas ideal. Aquesta preveu el comportament P-v-T amb bastanta exactitud, és a dir:

$$P \cdot V = R \cdot T \quad (\text{Eq. 3.1})$$

on R és la constant del gas.

Normalment, un gas ideal ha de ser una substància pura. No obstant això, l'aire és una barreja de gasos que obeeix l'equació de gas ideal en un ampli ventall de valor de temperatura i pressió. S'ha observat que aquesta equació funciona si la pressió no és massa alta o la temperatura massa baixa, ja que, a pressions baixes i temperatures altes la densitat del gas baixa i es comporta com un gas ideal. Molts gasos familiars com l'aire, nitrogen, oxigen, hidrogen, heli, argó, criptó e inclús algun més pesat com el diòxid de carboni, poden tractar-se com a gasos ideals amb un marge d'error insignificant.

A pressions més altes o temperatures més baixes, l'equació d'estat es complica. Cal tenir en compte el volum que agafen les molècules del gas, i que l'atracció de les molècules entre elles disminueix la pressió que exerceixen sobre el seu recipient. S'han proposat moltes equacions per representar les dades P-V-T amb més precisió que la llei de gasos ideal, però la majoria son empíriques i només algunes són d'ús ampli en la termodinàmica, l'enginyeria i camps físics relacionats. Dos equacions d'estat molt usades en enginyeria son les de Redliche-Kwong i Peng-Robinson. L'equació d'estat de Peng-Robinson és la que utilitza el codi de simulació flowsolverRHEA.

3.3. NIST

L'Institut Nacional d'Estàndards i Tecnologia (NIST) és una agència de de l'administració de tecnologia dels departament de Comerç dels Estats Units.

Des de la xarxa elèctrica intel·ligent i els registres sanitaris electrònics fins als rellotges atòmics, els nanomaterials avançats i els xips d'ordinador, innumbrables productes i serveis depenen d'alguna manera de la tecnologia, la mesura i els estàndards proporcionats per l'Institut Nacional d'Estàndards i Tecnologia.

Avui en dia, les mesures NIST admeten les tecnologies més petites fins a la més gran i complexa de creacions fetes per humans, des de dispositius a escala nanomètrica tan petits que desenes de milers poden cabre a l'extrem d'un sol cabell humà fins a gratacels resistents als terratrèmols i xarxes de comunicació globals .

Part de la missió del NIST és proveir a les indústries, acadèmies, governs i altres usuaris material de referència de patrons de més alta qualitat i valor metrollògic. Desenvolupen noves ferramentes per mesurar atributs crítics i proporcionar dades fidedignes entre altres. Per tal de comparar els resultats obtingut amb el flowsolverRHEA amb unes dades fiables, s'utilitza el Llibre del Web de Química del NIST, SRD 69. Allí s'obtenen les propietats termofísiques dels sistemes fluids.

4. FlowsolverRHEA

4.1. Necessitat de crear un solver en flux supercrític

La tecnologia dels microfluids ha crescut ràpidament les últimes dècades a causa de les seves altes relacions de superfície-volum, fluïdesa i capacitat de control i escales de longitud en l'ordre de microorganismes. Aquestes propietats encaixen perfectament a la biologia i la química en què l'alta precisió i els rendiments encaixen a la perfecció. L'aplicació de microfluids per aplicacions energètiques, que impliquen especialment la mescla i transferència de calor, ha sigut un gran entrebanc. Una de les principals raons és que l'energia global (el conjunt d'energia que hi ha en un sistema) és principalment un problema a gran escala en contrast amb els microcanals. Per tant, per tenir un impacte significatiu, les tecnologies d'energia microfluídica han d'oferir escalabilitat massiva o rellevant pels processos energètics que ja operen a escala. L'estudi i l'optimització de problemes tan complexos pot beneficiar-se de la utilització eficient de noves instal·lacions de computació d'alt rendiment (HPC) per afrontar un gran volum de càlculs i obtenir resultats en un període de temps raonable. Aquestes instal·lacions d'alt rendiment (HPC) utilitzen una potència de càlcul elevada per resoldre problemes complexos de ciència, enginyeria i gestió.

Molts dispositius de conversió d'energia, com motors de coets, funcionen a pressions que excedeixen la pressió crítica dels fluids involucrats. La pressió es troba per sobre de 10 MPa, per tant, més alta que la pressió crítica d'alguns fluids com oxigen ($P_c=5,04$ Mpa) o l'hidrogen ($P_c=3,39$ MPa). Per altra banda, l'interès per reduir les emissions de CO_2 i NO_x en els motors d'aeronaus i automòbils ha motivat l'augment de pressió dels fluids fins a superar la seva pressió crítica.

En quant al disseny i evolució de les turbines de vapor ha permès establir condicions d'operació superiors al punt crític, incrementant la potència generada i l'eficiència tèrmica amb relació a les centrals de generació d'energia que implementen turbines convencionals o subcrítiques.

L'objectiu és desenvolupar i utilitzar un marc computacional a exaescala per estudiar la utilització de fluids a alta pressió (supercrítics) per obtenir microfluids turbulents per aplicacions energètiques millorades. La computació a exaescala fa referència als sistemes de computació capaços de realitzar una mínim de 1 exaflop (10^{18}) u operacions de coma flotant per segon. Aquesta capacitat és una millora 1000 vegades superior al sistema de petaescala. Aquesta generació aporta capacitats a camps com l'exploració científica, d'intel·ligència artificial i l'anàlisi de dades.

Arribar a condicions supercrítiques d'alta pressió en dispositius microfluídics pot permetre una transició controlada bé entre règims laminars i turbulents que es poden aprofitar per superar les limitacions actuals per a un ampli ventall de cabals i mides de canals.

4.2. FlowsolverRHEA

El flow flowsolverRHEA ha sigut desenvolupat pel Departament de Mecànica de Fluids de la Universitat Politècnica de Catalunya. Des d'un principi ha estat dissenyat per ser un codi obert, reproduïble, fàcilment portable a diferents superordinadors i un pont per a la col·laboració entre acadèmia i indústria.

El codi està destinat a simulacions a gran escala d'alta fiabilitat en règims de turbulència a pressions supercrítiques en supercomputadors d'arquitectura híbrida. El flow solver, anomenat flowsolverRHEA, ja ha estat validat amb problemes canònics de problemes de flux de fluids, que involucren (proves separades, individuals) termodinàmica de gas ideal, flux delimitats per parets a microescala i turbulència isotròpica homogènia. Provat en nodes basats en CPU de BSC, i preparat per ser implementat en arquitectures híbrides a curt / mitjà termini.

El flowsolverRHEA està programat en C++ orientat a objectes, es basa en MPI i OpenACC per a la paral·lelització en CPUs i GPU, i utilitza HDF5 i Yaml per al tractament de dades i fitxers d'entrada/sortida.

4.3. Equació d'estat de Peng-Robinson

El flowsolverRHEA incorpora diferents equacions d'estat per realitzar els càlculs de flux. En aquest projecte s'estudiarà el funcionament del flowsolverRHEA utilitzant l'equació d'estat de Peng-Robinson.

L'equació d'estat de Peng-Robinson incorpora les millores que va fer Soave al model matemàtic de Redlich-Kwong. Aquest nou model té alguns avantatges significatius sobre els models d'equacions d'estat de dos paràmetres anteriors. Va ser desenvolupada per complir els següents objectius:

1. Els paràmetres havien de poder ser expressats en funció de les propietats crítiques i factor acèntric.
2. El model havia de ser raonablement precís a prop del punt crític, particularment per càlculs del factor de compressibilitat i la densitat líquida.
3. Les regles de mesclat no devien emprar més que un paràmetre sobre les interaccions binàries, que devien ser independents de la temperatura, pressió i composició.

4. L'equació devia ser aplicable a tots els càlculs de totes les propietats dels fluids en processos naturals de gasos.

L'equació d'estat és la següent:

$$P = \frac{RT}{v-b} - \frac{aa}{v^2+2bv+b^2} \quad (\text{Eq 4.1})$$

El factor ascèntric, omega ω , està relacionat amb la geometria de la molècula del gas i va definit per:

$$\omega = -1 - \log_{10} \left(\frac{p^{sat}}{P_c} \right)_{T/T_c=0.7} \quad (\text{Eq 4.2})$$

Les constants de Peng-Robinson estan determinades per:

$$k = 0.37464 + 1.54226\omega - 0.26993\omega^2 \quad (\text{Eq 4.3})$$

$$\alpha = \left(1 + k \left(1 - \sqrt{T/T_c} \right) \right)^2 \quad (\text{Eq 4.4})$$

La funció α es va obtenir linealitzant la relació de dades experimentals entre el punt d'ebullició normal i el punt crític.

$$a = 0.45723533 \frac{R^2 T_c^2}{P_c} \quad (\text{Eq 4.5})$$

$$b = 0.07779607 \frac{RT_c}{P_c} \quad (\text{Eq 4.6})$$

En aquesta equació, el paràmetre b és una correcció del volum mentre que el paràmetre a és un paràmetre d'interacció molecular. Aquestes constants depenen de la pressió i temperatura crítica del gas i , per tant, son constants en totes les condicions del gas.

La resolució de l'equació cúbica de Peng-Robinson es pot resoldre utilitzant una computadora o de forma analítica. Per facilitar la seva resolució, l'equació es pot transformar a la forma següent:

$$v^3 + \left(b - \frac{RT}{p} \right) v^2 + \left(\frac{aa}{p} - 3b^2 - 2 \frac{RT}{p} b \right) v + \left(b^3 + \frac{RT}{p} b^2 - \frac{aa}{p} b \right) = 0 \quad (\text{Eq 4.7})$$

$$v^3 + a_1 v^2 + a_2 v + a_3 = 0 \quad (\text{Eq 4.8})$$

La solució analítica ve donada per:

$$v^3 + a_1v^2 + a_2v + a_3 = 0 \quad (\text{Eq 4.9})$$

Que es transforma en:

$$x^3 + b_1x + b_2 = 0 \quad (\text{Eq 4.10}) \quad v = x - \frac{a_1}{3} \quad (\text{Eq 4.11})$$

$$b_1 = \frac{3a_2 - a_1^2}{3} \quad (\text{Eq 4.12}) \quad b_2 = \frac{2a_1^3 - 9a_1a_2 + 27a_3}{27} \quad (\text{Eq 4.13})$$

En la resolució de l'equació cúbica es poden donar 3 casos:

$$\frac{b_2^2}{4} + \frac{b_1^3}{27} > 0 \quad (\text{Eq 4.14})$$

En aquest cas l'equació tindrà una solució real i dos imaginàries.

$$\frac{b_2^2}{4} + \frac{b_1^3}{27} = 0 \quad (\text{Eq 4.15})$$

En aquest cas l'equació tindrà tres solucions reals, dos de les quals iguals.

$$\frac{b_2^2}{4} + \frac{b_1^3}{27} < 0 \quad (\text{Eq 4.16})$$

En aquest cas l'equació tindrà tres solucions reals diferents.

El primer cas és el més favorable i en centrarem en la solució d'aquest cas. En els altres dos casos també es pot arribar a una solució, però la seva resolució és molt més complexa. En aquests dos casos el flowsolverRHEA dona error, pel que durant el projecte només es treballarà per estar dintre del primer cas.

$$C = \sqrt[3]{-\frac{b_2}{2} + \sqrt{\frac{b_2^2}{4} + \frac{b_1^3}{27}}} \quad (\text{Eq 4.17})$$

$$D = \sqrt[3]{-\frac{b_2}{2} - \sqrt{\frac{b_2^2}{4} + \frac{b_1^3}{27}}} \quad (\text{Eq 4.18})$$

$$x = C + D \quad (\text{Eq 4.19})$$

$$v = x - \frac{a_1}{3} \quad (\text{Eq 4.20})$$

4.3.1. Exemple de càlcul amb l'equació d'estat Peng-Robinson

Per veure el funcionament de l'equació d'estat de Peng-Robinson, s'utilitza d'exemple el nitrogen a una pressió de 4Mpa i una temperatura de 160K. Les dades necessàries del Nitrogen per resoldre el problema es poden trobar a fitxer "substances_library_file", que es troba al flowsolverRHEA i també es pot consultar a [l'Annex A](#).

$$a = 0.45723533 \frac{8314^2 \cdot 126,129^2}{3395800} = 148211,380 \text{ m}^3/\text{kg} \cdot \text{mol}^2 \quad (\text{Eq 4.21})$$

$$b = 0.07779607 \frac{8314 \cdot 126,129}{3395800} = 0,024 \text{ m}^3/\text{kg} \cdot \text{mol} \quad (\text{Eq 4.22})$$

$$k = 0.37464 + 1.54226 \cdot 0,0372 - 0.26993 \cdot 0,0372^2 = 0,432 \quad (\text{Eq 4.23})$$

$$\alpha = \left(1 + 0,432 \left(1 - \sqrt{160/126,129} \right) \right)^2 = 0,894 \quad (\text{Eq 4.24})$$

$$a_1 = b - \frac{RT}{p} = -0,309 \quad (\text{Eq 4.25})$$

$$a_2 = \frac{\alpha a}{p} - 3b^2 - 2 \frac{RT}{p} b = 0,015 \quad (\text{Eq 4.26})$$

$$a_3 = b^3 + \frac{RT}{p} b^2 - \frac{\alpha a}{p} b = -0,00059 \quad (\text{Eq 4.27})$$

$$b_1 = \frac{3a_2 - a_1^2}{3} = -0,016 \quad (\text{Eq 4.28})$$

$$b_2 = \frac{2a_1^3 - 9a_1 a_2 + 27a_3}{27} = -0,0012 \quad (\text{Eq 4.29})$$

$$\frac{b_2^2}{4} + \frac{b_1^3}{27} = 1,87624E - 07 > 0 \quad (\text{Eq 4.30})$$

$$C = \sqrt[3]{-\frac{b_2}{2} + \sqrt{\frac{b_2^2}{4} + \frac{b_1^3}{27}}} = 0,100777986 \quad (\text{Eq 4.31})$$

$$D = \sqrt[3]{-\frac{b_2}{2} - \sqrt{\frac{b_2^2}{4} + \frac{b_1^3}{27}}} = 0,053970848 \quad (\text{Eq 4.32})$$

$$v = \frac{(C+D-\frac{a_1}{3})}{28,0134} = 0,00919 \text{ m}^3/\text{kg} \quad (\text{Eq 4.33})$$

$$\rho = \frac{1}{0,00919} = 108,75 \text{ kg/m}^3 \quad (\text{Eq 4.34})$$

Si es compara aquest valor amb el del NIST ($\rho = 105,23 \text{ kg/m}^3$) es pot calcular l'error:

$$\text{Error} = \frac{108,75 - 105,23}{105,23} 100 = 3,35\% \quad (\text{Eq 4.35})$$

4.3.2. Limitacions de l'equació

L'objectiu principal del projecte ha sigut fer les primeres simulacions i comparar-les per veure com de bé funciona el solver i les seves limitacions. Per buscar les seves limitacions s'han simulat diferents fluids a diferents rangs de pressions i temperatures. A pressions molt altes algunes propietats com la viscositat i la conductivitat tèrmica ja disten dels valors de referència del NIST, o directament el programa ja dóna algun error. Per tal de buscar les seves limitacions s'ha estudiat l'equació de Peng-Robinson, ja comentada en [l'apartat 4.3.1](#).

L'equació de Peng-Robinson és una equació cúbica, i per tant, pot tenir 3 solucions. Es defineixen les constants b_1 i b_2 i s'estudia quin és el resultat de la següent operació:

$$\frac{b_2^2}{4} + \frac{b_1^3}{27} \quad (\text{Eq 4.36})$$

Quan el resultat és més gran que 0 l'equació funcionarà bé, per tant, es busca treballar en aquesta zona. Per avaluar el resultat d'aquesta operació s'ha creat un petit programa amb Scilab. Introduint la pressió i un rang de temperatures el programa extreu una gràfica del resultat de l'operació en funció de la temperatura. Mentre el coeficient està per sobre de 0 l'equació funcionarà bé per aquesta pressió i temperatura.

Per cada pressió que s'ha volgut simular s'ha utilitzat el programa per extreure una gràfica amb el rang de temperatura en el que el solver funcionarà. A continuació es mostra un exemple d'una gràfica extreta a partir del programa:

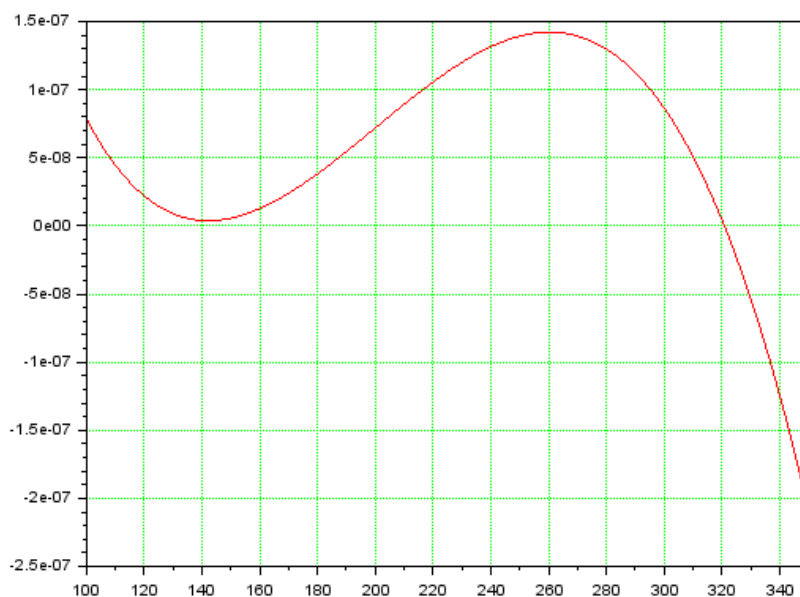


Figura 4-1. Coeficient del Nitrogen 6,79MPa en funció de la temperatura (Font: Josep Gorres, Elaboració pròpia)

La gràfica mostra el resultat del coeficient en funció de la temperatura. En aquest cas particular, el coeficient és positiu fins als 320K. A partir d'aquesta temperatura el coeficient passa a ser 0 en un punt i després ja és negatiu. El solver donarà un error.

4.4. Janaf-NASA 7-coeficients polinomials

Els polinomis es van mencionar per primera vegada com una forma per presentar propietats termoquímiques com la capacitat calorífica (C_p), entalpia (h), etc, en funció de la temperatura. El grup termodinàmic del centre Lewis de la NASA va dur un estudi exhaustiu fins trobar el famós polinomi de 7 coeficients de la NASA. Aquest 7 coeficients es poden utilitzar per calcular propietats fins als 6000K. Inicialment no més hi havia 7 coeficients, però després ho van ampliar a 14. Els primers 7 coeficients són per calcular des de 200K fins 1000K i els 7 restants són per calcular des de 1000K fins 6000K. El

resultat per 1000K serà el mateix tant si s'utilitzen els 7 primers o els 7 darrers. A continuació es mostren les fórmules per calcular la calor específica (C_p), l'entalpia (H_T) i l'entropia (S_T), a partir dels 7 coeficients i en funció de la temperatura.

$$\frac{C_p}{R} = a_1 + a_2T + a_3T^2 + a_4T^3 + a_5T^4 \quad (\text{Eq 4.37})$$

$$\frac{H_T}{RT} = a_1 + \frac{a_2T}{2} + \frac{a_3T^2}{3} + \frac{a_4T^3}{4} + \frac{a_5T^4}{5} + \frac{a_6}{T} \quad (\text{Eq 4.38})$$

$$\frac{S_T}{R} = a_1 \ln T + a_2T + \frac{a_3T^2}{2} + \frac{a_4T^3}{3} + \frac{a_5T^4}{4} + a_7 \quad (\text{Eq 4.39})$$

Cal senyalar que el valor H_T obtingut dels polinomis és l'entalpia de l'enginyeria definida com:

$$H_T = \Delta_f H_{298} + \int_{298}^T C_p dT \quad (\text{Eq 4.40})$$

Aquests paràmetres calculats a partir del polinomi de 7 coeficients es fan servir per calcular la part energètica com l'energia interna i la conductivitat tèrmica.

5. Procediment

5.1. Crear llibreria de substàncies

El primer pas abans d'iniciar les simulacions ha sigut crear la llibreria de substàncies. La llibreria de substàncies és un document de text on hi ha 6 substàncies amb les seves propietats i no cal haver de posar-les manualment en cada simulació. Aquestes substàncies son: Nitrogen, Oxigen, Hidrogen, Metà, Dodecà i Diòxid de Carboni. La llibreria de substàncies es troba a [l'Annex A](#).

Les propietats que incorpora la llibreria son: R específica, gamma, pes molecular, factor acèntric, pressió crítica, temperatura crítica, volum molar crític i els coeficient de la NASA. Per obtenir aquestes propietats s'han utilitzat tres fonts d'informació: el NIST, el thermochemical data base i The Properties of Gases and Liquids. Aquestes tres fonts d'informació es poden consultar a la [Bibliografia](#).

A partir de les taules l'apèndix A de propietats gases líquids s'obté la massa molecular.

No.	Formula	Name	CAS #	Mol. Wt.	Tfp, K	Tb, K	Tc, K	Pc, bar	Vc, cm ³ /mol	Zc = PcVc/RTc	Omega
449	H ₄ N ₂	hydrazine	302-01-2	32.045	274.68	386.65	653.01	147.00	101.10	0.282	
450	He	helium	7440-59-7	4.003	2.15	4.30	5.19	2.27	57.30	0.301	-0.390
451	He	helium-3	14762-55-1	3.017	1.01	3.33	3.31	1.14	72.50	0.300	-0.480
452	I ₂	iodine	7553-56-2	253.809	386.76	457.56	819.00		155.00		
453	Kr	krypton	7439-90-9	83.800	115.77	119.74	209.40	55.00	91.20	0.288	
454	NO	nitrogen monoxide (nitric oxide)	10102-43-9	30.006	109.51	121.38	180.00	64.80	58.00	0.251	0.582
455	N ₂	nitrogen	7727-37-9	28.014	63.15	77.35	126.20	33.98	90.10	0.289	0.037
456	N ₂ O	dinitrogen oxide (nitrous oxide)	10024-97-2	44.013	182.33	184.67	309.60	72.55	97.00	0.273	
457	N ₂ O ₄	dinitrogen tetroxide (nitrogen dioxide)	10544-72-6	92.011	261.95	302.22	431.01	101.00			1.007

Figura 5-1. Propietat Nitrogen (Font: The Properties of Gases and Liquids, McGRAW-HILL 5a edició)

El factor acèntric, temperatura crítica, pressió crítica i volum molar crític s'obtenen a la pàgina del NIST.

Additional fluid properties

Critical temperature (T_c)	126.192 K
Critical pressure (P_c)	3.3958 MPa
Critical density (D_c)	313.300 kg/m ³
Acentric factor	0.0372
Normal boiling point	77.355 K
Dipole moment	0.0 Debye

Figura 5-2. Propietat dels Nitrogen (Font: NIST)

La R específica es calcula a partir de la massa molecular i la gamma es calcula a partir del calor específic, que aquest s'obté del thermochemical data base.

Compound	Mol. Wgt.	$\Delta_f H_{298}$ kJ/mol	$\Delta_f H_0$ kJ/mol	\pm kJ/mol	C_{p298} J/mol/K	S_{298} J/mol/K	$H_{298}-H_0$ kJ/mol	
NO	30.00614	91.271	90.767		29.862	210.748	9.179	†
NO+	30.00559	990.807	982.137		29.123	198.234		†
NOCL	65.45884	52.524	54.425	±0.5	44.623	261.590	11.364	†
NOF	49.00454	-65	-62.633	±2.0	41.530	248.224	10.720	†
NOF3	87.00135	-187	-178.78	±7.	68.067	277.731	13.698	†
NO2	46.00554	34.193	37.0	±0.5	37.177	240.171	10.208	†
NO2-	46.00609	-200.036			37.215	236.241		†
NO2CL	81.45824	12.5	17.901	±1.	53.246	272.128	12.205	†
NO2F	65.00394	-109	-102.92	±20	48.999	259.287	11.347	†
NO3	62.00494	74.628	81.024	±0.69	46.935	252.623	10.959	†
NO3-	62.00549	-310.78	-298.0		44.724	245.638		†
NO3F	81.00334	15			66.958	293.171		†
N2 REFERENCE ELEMENT	28.01348	0	0		29.124	191.607	8.670	††

Figura 5-3. Propietats del Nitrogen (Font: Thermochemical Database, The University of Chicago)

$$R_{specific} = \frac{8,314}{0,027814} = 296,80 \text{ J/kgK} \quad (\text{Eq 5.1})$$

$$C_p = \frac{29,124}{0,028014} = 1039,62 \text{ J/kgK} \quad (\text{Eq 5.2})$$

$$k = \frac{c_p}{c_p - R} = \frac{1039,62}{1039,62 - 296,80} = 1,4 \quad (\text{Eq 5.3})$$

Els 7 coeficients de la NASA s'obtenen a partir de thermochemical data base.

```

7727-37-9
N2 HF298= 0.0 KJ REF=TSIV Max Lst Sq Error Cp @ 6000 K 0.29%
N2 REF ELEMENT G 8/02N 2. 0. 0. 0.G 200.000 6000.000 A 28.01340 1
2.95257637E+00 1.39690040E-03-4.92631603E-07 7.86010195E-11-4.60755204E-15 2
-9.23948688E+02 5.87188762E+00 3.53100528E+00-1.23660988E-04-5.02999433E-07 3
2.43530612E-09-1.40881235E-12-1.04697628E+03 2.96747038E+00 0.00000000E+00 4

```

Figura 5-4. Coeficient de la NASA per el Nitrogen (Font: Font: Thermochemical Database, The University of Chicago)

Un cop editat el fitxer de les substàncies, les substàncies noves s'han d'afegir al fitxer de "substances_selection". Aquest fitxer es troba a [l'Annex A](#).

5.2. Propietats NIST

A continuació, s'han descarregat del NIST les propietats de les substàncies per comparar-les en les simulacions. Un cop realitzades les primeres simulacions s'ha anat modificant el programa de postprocessat per obtenir gràfiques de forma més ràpida. A continuació es mostra el procediment per descarregar les dades i modificar el fitxer perquè sigui compatible amb el programa de postprocessat del flowsolverRHEA.

Por favor siga los pasos siguientes para seleccionar los datos requeridos.

1. Por favor escoja la especie de interés:

Nitrógeno

2. Por favor escoja las unidades que desea usar:

Temperatura

Kelvin Celsius Fahrenheit Rankine

Presión

MPa bar atm. torr psia

Densidad

mol/l mol/m³ g/ml kg/m³ lb-mole/ft³ lbm/ft³

Energía

kJ/mol kJ/kg kcal/mol Btu/lb-mole kcal/g Btu/lbm

Velocidad

m/s ft/s mph

Viscosidad

μPa*s Pa*s cP lbm/ft*s

Tensión superficial*

N/m dyn/cm lb/ft lb/in

3. Escoja los datos deseados:

Tipo de datos

Propiedades Isotérmicas

Propiedades Isobáricas

Propiedades isocóricas

Propiedades de saturación — Incrementos de temperatura

Propiedades de saturación — Incrementos de presión

4. Por favor seleccione la convención del estado patrón deseada:

Standard state convention

Por definición para fluidos

Convención de punto de ebullición normal

Convención ASHRAE

Convención IIR

5. [Pulse para continuar](#)

Figura 5-5. Selecció Nitrogen NIST (Font: NIST)

Isobaric properties for Nitrogen

This option will supply data on a constant pressure curve over the specified temperature range. Values should not extend outside the minimum and maximum values given. Calculations are limited to a maximum of 601 data points; increments resulting in a larger number of points will be adjusted upward to limit the number of points computed.

- Enter pressure in selected units:
 (Acceptable range: 0.0 to 2200.0000 MPa)
- Enter temperature range and increment in selected units:

T _{Low}	<input type="text" value="0"/>	
T _{High}	<input type="text" value="1000"/>	(max value: 2000.0 K)
T _{Increment}	<input type="text" value="1"/>	
- * The minimum temperature limit is the highest of the following values:
 - 63.151 K
 - The temperature at which a density of 867.21 kg/m³ is reached.
- Check here if you want to use the interactive display (requires JavaScript and HTML 5 canvas capable browser)
- Number of digits to be displayed in tables (does not effect accuracy of computations):
-

Figura 5-6. Dades d'entrada (Font: NIST)

Isobaric Properties for Nitrogen

- Fluid Data
- Auxiliary Data
- References
- Additional Information
- Important Information About This Data
- Notes
- Other Data Available:
 - View data in HTML table.
 - Download data
 - Main NIST Critical Point
 - Recommendations
 - Fluid data for

The following adjustments are available:

- The specified range
- The specified increment

Available data: [] of points calculated.

Fluid Data

Figura 5-7. Descàrrega de les dades (Font: NIST)

El fitxer es guarda en un document de text. A aquest document se li han de fer unes petites modificacions perquè sigui compatible amb el postprocessat. La funció del postprocessat és fer gràfiques que comparin els resultats del solver amb les del NIST. Per editar els documents de text de forma àgil s'utilitza Notepad++. A continuació es mostra el procés.

- Afegir el símbol “#” a la primera frase per tal que python no intenti executar el text.
- Substituir l'espai en blanc entre columnes per una coma. Mitjançant l'opció reemplaçar que es troba al menú buscar.

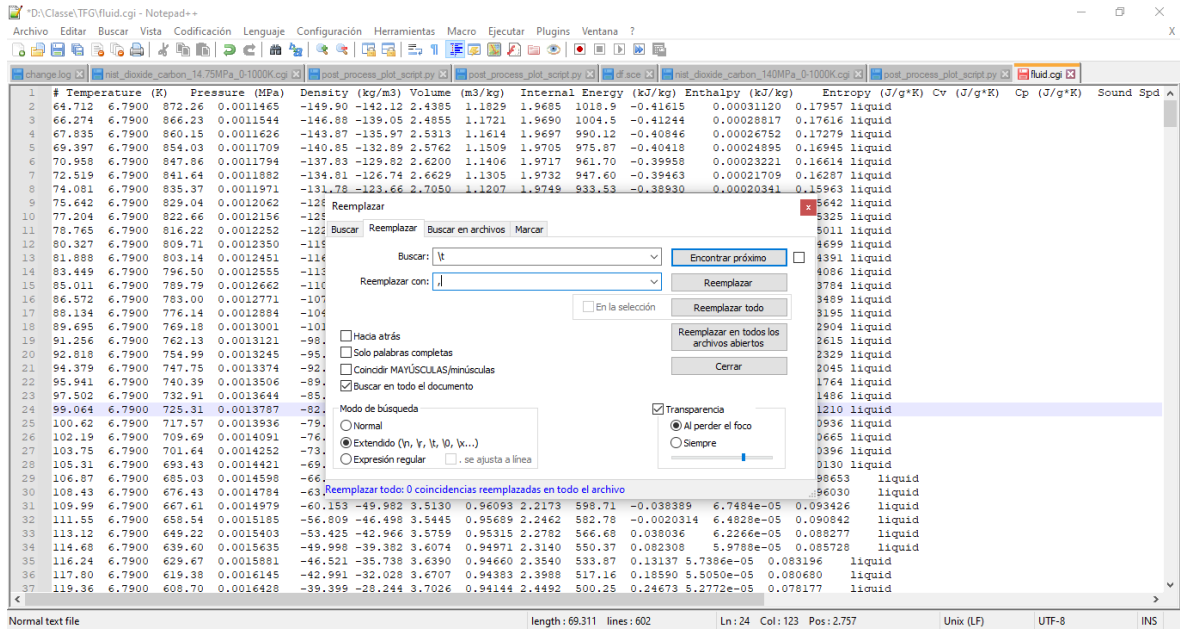


Figura 5-8. Edició fitxer de dades (Font: Josep Gorres, Elaboració pròpia)

3. Substituir la paraula “líquid” per un espai en blanc i repetir la mateixa operació per la paraula “supercritical”.

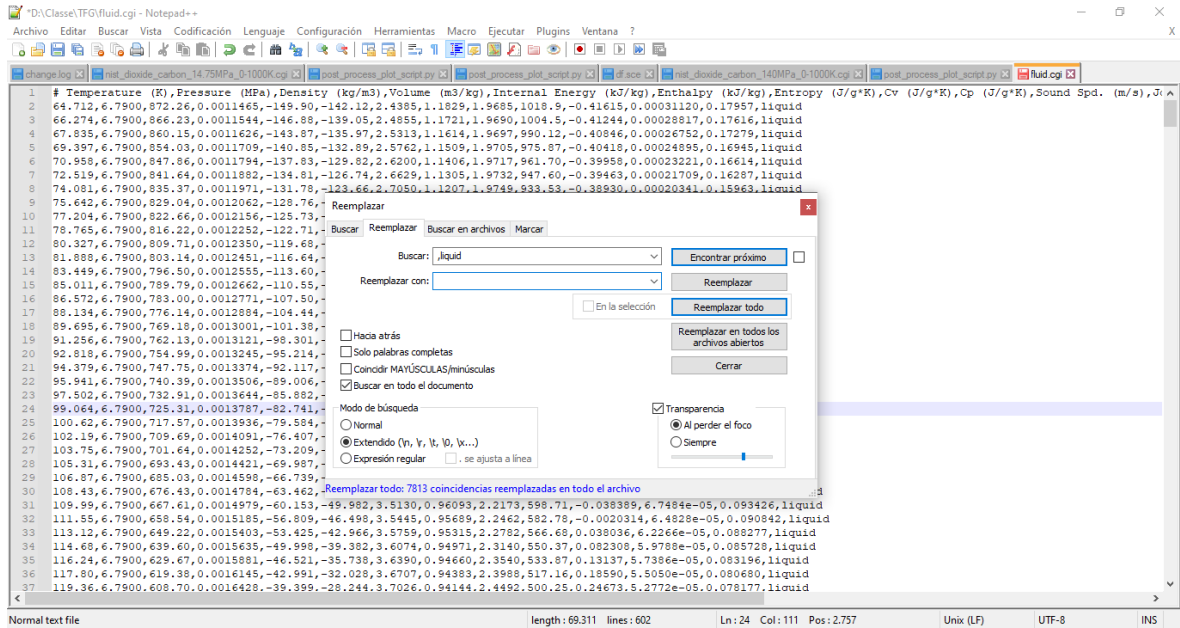


Figura 5-9. Edició fitxer de dades (Font: Josep Gorres, Elaboració pròpia)

4. Canviar el nom del fitxer per “nist_nomsustància_pressióMPa_0-1000K”, un exemple podria ser: nist_nitrogen_339.58MPa_0-1000K.

5.3. Postprocessat

El document de postprocessat és un executable que funciona amb el llenguatge de programació Python. La funció d'aquest fitxer és representar en una mateixa gràfica els resultats de la simulació i les propietats extrems del NIST, per tal de comparar-los i avaluar el funcionament del solver.

Les 5 propietats que es representen són la densitat, l'energia interna, la velocitat del so, la conductivitat tèrmica i la viscositat. Totes aquestes propietats es representen en funció de la temperatura.

A l'inici del projecte ja es partia d'un postprocessat que obria els fitxers dels resultats i feia les gràfiques. Els límits dels eixos verticals s'havien de definir manualment i en cada simulació podien variar depenent de les condicions. Les gràfiques sortien en format eps. Per visualitzar aquest tipus de fitxer és necessari una aplicació específica. El postprocessat original es pot consultar a [l'annex A](#).

L'objectiu de modificar el postprocessat és que defineixi de forma automàtica els límits dels eixos verticals per no haver-los de modificar en cada simulació i així poder realitzar diferents simulacions de forma més àgil. També que les gràfiques surtin en format eps i png per visualitzar-les de forma més fàcil. Aquest es pot consultar a [l'Annex A](#).

Es parteix de dos documents de dades per realitzar les gràfiques. El document del NIST en format CSV i el document del flowsolverRHEA en format H5. En la majoria de casos el document del NIST té dades de 0K fins 1000K, mentre que el flowsolverRHEA només té les dades de l'interval que s'ha simulat. El primer pas és trobar en quina posició del fitxer del NIST comencen les dades que corresponen a l'interval de la simulació. Aquesta operació es fa mitjançant un bucle amb el While. A continuació es mostra aquesta part del programa:

```
t_min = round(T_data[0], 0)      # Temperature [K]
t_max = T_data[-1]             # Temperature [K]

posN1 = 0
while T_nist[posN1] < t_min:
    posN1 += 1

print(posN1)

posN2 = 0
while T_nist[posN2] < t_max:
    posN2 += 1

print(posN2)

posD1 = 0
while T_data[posD1] < t_min:
    posD1 += 1
```

```
print(posD1)

posD2 = 0
while T_data[posD2] < t_max:
    posD2 += 1

print(posD2)
```

Les variables t_{\min} i t_{\max} , corresponen a la temperatura mínima i màxima de l'interval de la simulació. Per defecte el fitxer representarà en l'interval de temperatures de la simulació, però en cas que es vulgui representar un altre interval es pot modificar. Un cop s'ha trobat la posició en què el NIST correspon a l'interval de la simulació, el següent pas és trobar els mínims i màxims de tots els valors.

Trobar els mínims i màxims ja és més complex, perquè poden no coincidir els del NIST amb el del flowsolverRHEA. La forma de trobar-los pot variar en funció de la substància i la pressió, però normalment se sol fer de la mateixa forma. A continuació es mostra aquesta part de l'oxigen a una pressió de 82Mpa.

```
x= (rho_data[posD1], rho_data[posD2], rho_nist[posN1], rho_nist[posN2])
rho_min= round(min(x)-0.1*abs(max(rho_data)-min(rho_data)), 0)
rho_max= round(max(x)+0.1*abs(max(rho_data)-min(rho_data)), 0)
sos_min= round(min(sos_data)-0.1*(max(sos_data)-min(sos_data)), 0)
sos_max= round(max(sos_data)+0.1*(max(sos_data)-min(sos_data)), 0)
y= (mu_data[posD1], mu_data[posD2], mu_nist[posN1], mu_nist[posN2])
mu_min= round(min(y)-0.1*(max(mu_data)-min(mu_data)), 5)
mu_max= round(max(y)+0.1*(max(mu_data)-min(mu_data)), 5)
z= (kappa_data[posD1], kappa_data[posD2], kappa_nist[posN1], kappa_nist[posN2])
kappa_min= round(min(kappa_data)-0.1*(min(kappa_data)), 2)
kappa_max= round(max(kappa_data)+0.1*(max(kappa_data)-min(kappa_data)), 2)
```

En el primer cas la "x" busca la posició inicial i final del flowsolverRHEA i el NIST. Per trobar els extrems s'usa el mínim i màxim del NIST i el flowsolverRHEA. Aquesta opció funciona bé quan els mínims i màxims es troben a l'inici o el final de la corba, però si es troba pel mig pot fallar. L'avantatge és que té en compte els mínims i màxims del NIST i el flowsolverRHEA. També se li afegeix o es resta un 10% de la diferència entre el màxim i el mínim perquè la corba no quedi tan ajustada als límits de la gràfica.

En el segon cas només té en compte les dades del flowsolverRHEA. L'avantatge és que té en compte els mínims i màxims de tota la corba. L'inconvenient és que no té en compte les dades del NIST i si estan molt desfasades pot no sortir la corba dintre la gràfica.

Com ja s'ha dit anteriorment aquests paràmetres s'han modificat en funció de la substància i la pressió.

A continuació es mostra com es defineix la gràfica de l'energia interna respecte a la temperatura, perquè és un cas particular. En aquesta gràfica s'ha de fer un "offset" perquè la corba del flowsolverRHEA coincideixi amb la del NIST, si no estaria desfasada.

```

#### Plot internal energy vs. temperature

posN = 0
while T_nist[posN] < t_min:
    posN += 1

print(posN)

posD = 0
while T_data[posD] < t_min:
    posD += 1

print(posD)

offsetE=e_nist[posN]-e_data[posD]/1.0e3;

E_min= round(E_data[0]/1.0e3+offsetE-abs(0.1*(abs(E_data[-1]/1.0e3)-abs(E_data[0]/1.0e3))), 0)
E_max= round(E_data[-1]/1.0e3+offsetE+abs(0.1*(abs(E_data[-1]/1.0e3)-abs(E_data[0]/1.0e3))), 0)

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, e_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'$\text{NIST}$' )
plt.plot( T_data, e_data/1.0e3 + offsetE, linestyle = '--', color = 'firebrick', label = r'$\text{RHEA}$' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 20 ) )
plt.xscale( 'log' )
plt.xlabel( r'$T \text{ \thinspace } \text{K}$' )
plt.ylim( E_min, E_max )
plt.yticks( np.arange( E_min, E_max+ 10, 100 ) )
plt.yscale( 'log' )
plt.ylabel( r'$e \text{ \thinspace } \text{kJ/kg}$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper left' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'internal_energy_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'internal_energy_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

```

Aquesta operació és molt similar a la que es fa per trobar els mínims i màxims. Consisteix en trobar la posició en què coincideix les dades del flowsolverRHEA i el NIST. Un cop trobada aquesta posició se li resta el valor del flowsolverRHEA al NIST, per trobar la diferència. Un cop trobada la diferència se li suma al flowsolverRHEA per coincidir amb el NIST. També es divideix el flowsolverRHEA entre 1000 per passar-ho a kJ.

També hi ha un altre programa de postprocessat, que la seva funció és representar en una mateixa gràfica les dades del NIST, el resultat del flowsolverRHEA amb l'equació de Peng-Robinson i el resultat del flowsolverRHEA amb l'equació de gas ideal. El programa és el mateix que l'anterior, però afegint una tercera corba a la gràfica. Es pot consultar a [l'Annex A](#).

5.4. Limitació Peng-Robinson

Com ja s'ha comentat a l'apartat [4.3.2](#), l'equació de Peng-Robinson no funciona sempre bé. La implementació de l'equació de Peng-Robinson només té en compte el cas d'una solució real, per la resta de casos no funciona bé. En un cas obté 3 solucions, reals de les quals 2 son iguals, i en l'altre cas obté 3 solucions reals diferents. El cas que té en compte, és quan obté 1 solució real i 2 imaginàries. Aquesta solució real és el resultat correcte.

Pel que s'ha creat un programa amb Scilab. En un primer moment només es volia estudiar si el resultat de coeficient era positiu o negatiu per saber si l'equació funcionaria en aquelles condicions. Però ja que s'havia creat el programa es va aprofitar per calcular la densitat i comparar la gràfica amb el resultat del flowsolverRHEA, per saber si aquest programa calculava correctament els paràmetres.

El programa utilitza un bucle "for" per calcular els paràmetres entre la temperatura inicial i final. Totes les dades de les substàncies ja estan introduïdes, només cal modificar el rang de temperatures i la pressió. A [l'Annex B](#) es troben els programes per cada substància. A continuació, es mostra l'exemple de l'oxigen:

```
function RepresentaBB()
```

```
Tc=154.58  
Pc=5043000  
omega=0.0222  
Tini=100  
Tfi=250
```

```
CalculBB(Tc,Pc,omega,Tini,Tfi,82,"Res1.txt","r-")
```

```
endfunction
```

```
function CalculBB(Tc, Pc, omega, Tini, Tfi, P, nomRes, colGraf)
```

```
R=8314  
p=P*1000000
```

```
a=0.45723553*R^2*Tc^2/Pc  
b=0.07779607*R*Tc/Pc
```

```
kappa=0.37464+1.5422*omega-0.26993*omega^2
```

```
pos=1
```

```

for T=Tini:1:Tfi,
    alfa=(1+kappa*(1-(T/Tc)^0.5))^2
    a1= b-R*T/p
    a2= alfa*a/p-3*b^2-2*R*T*b/p
    a3= b^3+R*T*b^2/p-alfa*a*b/p

    b1= (3*a2-a1^2)/3
    b2= (2*a1^3-9*a1*a2+27*a3)/27

    T_vec(pos)=T
    BB_vec(pos)=b2^2/4+b1^3/27
    CC=nthroot((-b2/2+(BB_vec(pos))^0.5),3)
    DD=nthroot((-b2/2-(BB_vec(pos))^0.5),3)
    C_vec(pos)=CC
    D_vec(pos)=DD

    vol_vec= (CC+DD-a1/3)/31.999
    den_vec(pos)=1/vol_vec;
    pos=pos+1
end

//Per representar el valor del coeficient
scf(1)
plot(T_vec,BB_vec,colGraf)
xlabel(" Temperatura (K) ");
ylabel(" Coeficient ((m^3/kg·mol)^6) ");
xgrid(3)
matRes=[T_vec,BB_vec]
fprintfMat(nomRes+"_coef",matRes)

//Per representar el valor de volum específic
scf(2)
plot(T_vec,den_vec,colGraf)
xlabel(" Temperatura (K) ");
ylabel(" Densitat (kg/m^3) ");
xgrid(3)
matRes2=[T_vec,den_vec]
fprintfMat(nomRes+"_den",matRes2)

endfunction

```

A continuació, es mostren les dos gràfiques que s'obtenen del programa en aquest exemple en concret. La segona gràfica, que mostra la densitat, es pot comparar amb la del flowsolverRHEA per veure que el resultat és el mateix. Aquesta gràfica es pot consultar a [l'Annex C](#).

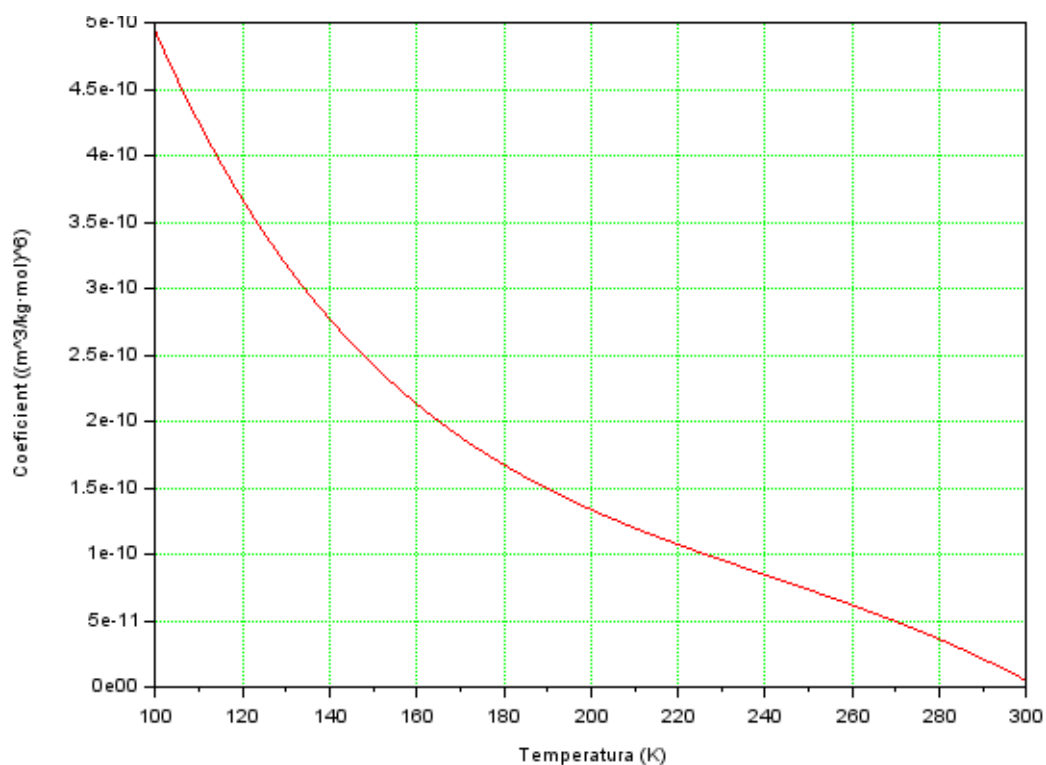


Figura 5-10. Coeficient de l'oxigen a 82MPa en funció de la temperatura. (Font: Josep Gorres, Elaboració pròpia)

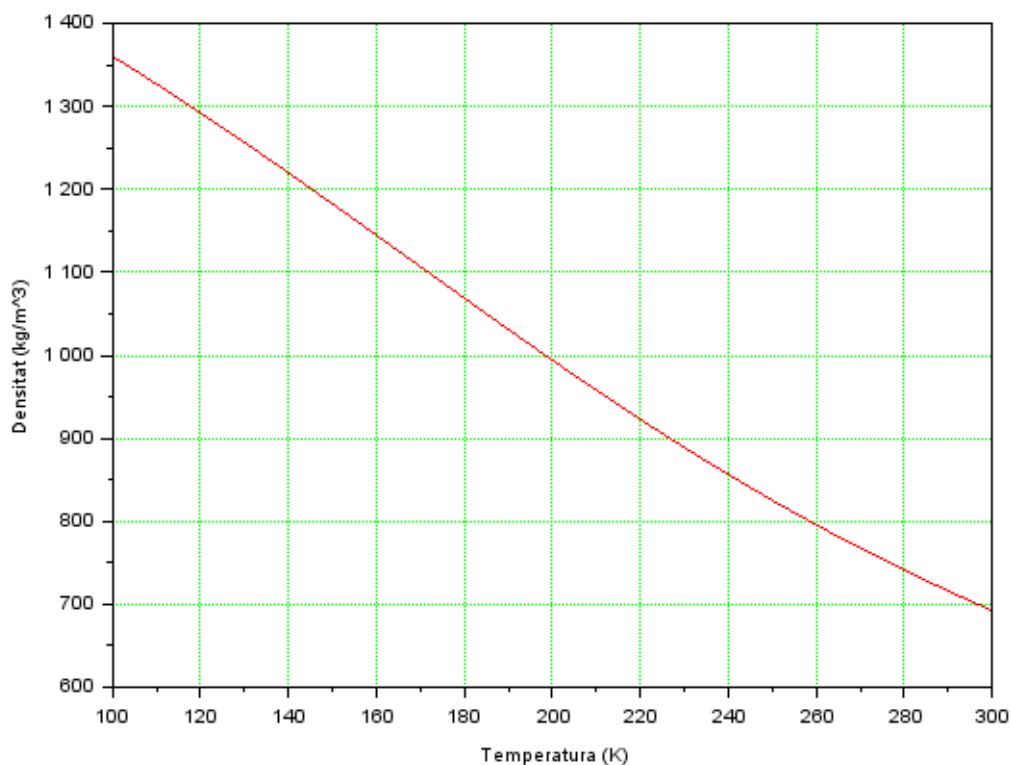


Figura 5-11. Densitat de l'oxigen a 82 MPa en funció de la temperatura. (Font: Josep Gorres, Elaboració pròpia)

5.5. Interval de simulació

El principal objectiu del projecte és realitzar les primeres simulacions amb el codi de simulació flowsolverRHEA per estudiar el seu funcionament i on comença a fallar. S'han seleccionat 6 substàncies per estudiar-les a diferents pressions i temperatures.

Cada substància s'ha simulat a 5 pressions diferents, i s'han tingut en compte dos criteris per seleccionar les temperatures i pressions.

- La pressió: en un principi es va recomanar estudiar la pressió en funció de la pressió crítica de la següent forma: $2 \cdot P_c$, $5 \cdot P_c$, $10 \cdot P_c$, $50 \cdot P_c$, $100 \cdot P_c$. La majoria de substàncies només s'ha pogut seguir aquest criteri fins $10 \cdot P_c$. perquè a pressions més elevades el coeficient ja era negatiu, i per tant, l'equació de Peng-Robinson ja no funciona. Per buscar quina és la pressió més elevada a la que l'equació de Peng-Robinson funciona, s'ha utilitzat el programa de Scilab. S'ha anat augmentant gradualment la pressió fins trobar la pressió màxima en que funciona. Un cop trobada aquesta pressió s'ha establert com la pressió més elevada a la que es simula i s'ha establert una altra pressió intermèdia entre $10 \cdot P_c$ i la pressió màxima. Les gràfiques de cada substància on es veu fins quina temperatura es pot simular a una pressió determinada cada substància es troben a [l'Annex B](#).

- La temperatura: el rang de temperatura que se simula cada pressió és al voltant del que podem considerar canvi de fase quan ens trobem per sobre del punt crític. Normalment s'estableix un interval de 100K al voltant d'aquest canvi de fase. Tenint en compte que no superi la temperatura en què funcionarà l'equació. Per establir aquest rang de temperatures s'ha consultat el NIST on es pot veure el canvi de fase a la gràfica.

A continuació, es mostra una taula amb les pressions i rang de temperatures per a cada substància.

Substància	Nitrogen	Diòxid de carboni	Oxigen	Dodecà	Hidrogen	Metà
Pressió crítica	3,3958	7,3773	5,0430	1,8170	1,2964	4,5992
Pressió 1 (MPa)	6,79	14,75	10,09	3,63	2,59	9,19
Rang temperatura 1 (K)	100 - 200	250 - 350	100 - 200	600 - 700	20 - 100	150 - 250
Pressió 2 (MPa)	16,97	36,89	25,22	9,09	6,48	22,99
Rang temperatura 2 (K)	100 - 200	250 - 350	100 - 200	600 - 700	20 - 100	150 - 250
Pressió 3 (MPa)	33,95	73,77	50,43	18,17	12,96	45,99
Rang temperatura 3 (K)	100 - 200	250 - 350	100 - 200	600 - 700	20 - 100	150 - 250
Pressió 4 (MPa)	60	140	65	50	20	80
Rang temperatura 4 (K)	100 - 200	250 - 350	100 - 200	600 - 700	20 - 90	150 - 250
Pressió 5 (MPa)	100	200	82	-	30	100
Rang temperatura 5 (K)	100 - 130	260 - 340	100 - 200	-	25 - 60	150 - 250

Taula 5-1. Rangs de temperatures i pressions per cada substància (Font: Josep Gorres, Elaboració Pròpia)

5.6. Comparació Peng-Robinson amb Gas Ideal

El flowsolverRHEA pot seguir diferents models per realitzar els càlculs. Aquest projecte es basa en estudiar el funcionament del flowsolverRHEA amb l'equació de Peng-Robinson. L'equació d'estat de gas ideal funciona bé, però a pressions més altes ja perd precisió.

Per tant, un bon estudi inicial és comparar els resultats de Peng-Robinson amb el gas ideal. Per comparar-les s'ha utilitzat el flowsolverRHEA amb les dues equacions i s'han comparat els resultats. Primer s'han simulat les substàncies amb les mateixes condicions en els dos models. Un cop obtinguts els resultats s'ha utilitzat el programa de postprocessat. Al programa de postprocessat inicial se li ha afegit una tercera corba, que és la del gas ideal. D'aquesta forma es pot veure visualment els resultats d'aquests dos models respecte al NIST.

6. Funcionament

El flowsolverRHEA està instal·lat en una computadora amb sistema operatiu Linux situada físicament al campus de la EEBE. Aquesta computadora s'ha utilitzat a distància mitjançant dues aplicacions: WinSCP i putty.

El WinSCP és un software per fer la transferència de fitxer entre el terminal i l'ordinador des d'on es treballa. El putty és un software per utilitzar la computadora i executar el solver.

6.1. Configuració fitxers

6.1.1. Configuration_file.yaml

En aquest fitxer es configura l'equació d'estat que utilitza el solver, que pot ser "PENG ROBINSON" o "IDEAL GAS". S'indica el nom de la substància per simular i la ruta d'accés a la llibreria de substàncies. La ruta d'accés a la llibreria dependrà de com estiguin repartides les carpetes. En aquest cas particular, el fitxer de substàncies no està a la mateixa carpeta que el flowsolverRHEA. El programa utilitza 3 cops l'operació "../", aquesta operació serveix per tirar endarrere a les carpetes. Un cop ha tirat endarrere 3 carpetes, ja indica el nom de la carpeta on es troba la llibreria de substàncies i finalment el nom del fitxer.

```

# RHEA's CONFIGURATION FILE (YAML LANGUAGE)
#
# 1D high-pressure framework test (Nitrogen, 4MPa, 100-200K):
# P. J. Linstrom and W. G. Mallard.
# Chemistry WebBook, NIST Standard Reference Database.
# http://webbook.nist.gov/chemistry/fluid/
---

##### FLUID & FLOW PROPERTIES #####
fluid_flow_properties:
  # Thermodynamic models: IDEAL_GAS (provide R_specific, gamma), STIFFENED_GAS (provide R_specific, gamma, P_inf, e_0, c_v),
  # PENG_ROBINSON (provide R_specific, molecular_weight, acentric_factor, critical temperature, critical pressure, critical molar volume, NASA 7-coefficient polynomial
  thermodynamic_model: 'PENG_ROBINSON'
  # Thermodynamic model
  #R_specific: 296.80
  # Specific gas constant [J/(kg.K)]
  #gamma: 1.4
  # Heat capacity ratio (ideal-gas) [-]
  #P_inf: 0.0
  # Pressure infinity (liquid stiffness) [Pa]
  #e_0: 0.0
  # Internal energy zero point [J/kg]
  #c_v: 0.0
  # Specific isochoric heat capacity [J/(kg.K)]
  #molecular_weight: 0.0280134
  # Molecular weight [kg/mol]
  #acentric_factor: 0.0372
  # Acentric factor [-]
  #critical_temperature: 126.192
  # Critical temperature [K]
  #critical_pressure: 3395800.0
  # Critical pressure [Pa]
  #critical_molar_volume: 0.000089412
  # Critical molar volume [m3/mol]
  #NASA_coefficients: [2.95257637, 0.0013969004, -0.000000492631603, 0.00000000078601019, -0.00000000000004607552, -923.948688, 5.87188762, 3.53100528, -0.0001236609]
  # Transport coefficients models: CONSTANT (provide mu, kappa),
  # HIGH_PRESSURE (provide molecular weight, critical temperature, critical molar volume, acentric factor, dipole moment, association factor, NASA 7-coefficient polyn
  transport_coefficients_model: 'HIGH_PRESSURE'
  # Transport coefficients model
  #mu: 0.0
  # Dynamic viscosity [Pa.s]
  #kappa: 0.0
  # Thermal conductivity [W/(m.k)]
  #dipole_moment: 0.0
  # Dipole moment [D]
  #association_factor: 0.0
  # Association factor [-]
  # Substances: NITROGEN
  # Optional. Define substance to read from library file: R_specific, gamma, P_inf, e_0, c_v, molecular_weight, acentric_factor, critical_temperature, critical_pressur
  substance_name: 'NITROGEN'
  # Substance [-]
  substances_library_file: '.././././src/substances_library_file.yaml'
  # Substances library file [-]

```

Figura 6-1. Configuration_file.yaml (Font: Josep Gorres, Elaboració pròpia)

6.1.2. MyRHEA.cpp

En aquest fitxer és on s'introdueixen les condicions inicials de la simulació, en aquests estudis son la pressió(Pa) i temperatura(K).

```
#include "myRHEA.hpp"
using namespace std;

////////// myRHEA CLASS //////////

void myRHEA::setInitialConditions() {

    /// IMPORTANT: This method needs to be modified/overwritten according to the problem under consideration

    /// All (inner, halo, boundary): u, v, w, P and T
    for(int i = topo->iter_common[_ALL][_INIX_]; i <= topo->iter_common[_ALL][_ENDX_]; i++) {
        for(int j = topo->iter_common[_ALL][_INIY_]; j <= topo->iter_common[_ALL][_ENDY_]; j++) {
            for(int k = topo->iter_common[_ALL][_INIZ_]; k <= topo->iter_common[_ALL][_ENDZ_]; k++) {
                u_field[I1D(i,j,k)] = 0.0;
                v_field[I1D(i,j,k)] = 0.0;
                w_field[I1D(i,j,k)] = 0.0;
                P_field[I1D(i,j,k)] = 6.79e6;
                T_field[I1D(i,j,k)] = 100.0*mesh->x[i] + 100.0;
            }
        }
    }

    /// Update halo values
    u_field.update();
    v_field.update();
    w_field.update();
    P_field.update();
    T_field.update();
};

void myRHEA::calculateSourceTerms() {

    /// IMPORTANT: This method needs to be modified/overwritten according to the problem under consideration
    <
```

Figura 6-2. MyRHEA.cpp (Font Josep Gorres, Elaboració pròpia)

En aquest exemple l'interval de temperatures és de 100K a 200K, però si es vol fer un interval de 250K a 350K s'ha d'expressar de la següent forma: 100.0*mesh->x[i] + 250.0

6.1.3. post_process_plot_script.py

En aquest fitxer només s'ha de canviar el nom del document csv en la linea Open NIST file, pel nom corresponent a la simulació. En alguns casos s'hauria de modificar els límits dels eixos verticals per ajustar millor les gràfiques.

```
<t( 'nist_nitrogen_33.95MPa_0-1000K.csv', delimiter=',', unpack = 'True' )
```

Figura 6-3. Post_process_plot_script.py (Font Josep Gorres, Elaboració pròpia)

6.2. Execució fitxers

El procediment consisteix en accedir a la carpeta mitjançant la comanda “cd”, utilitzar la comanda “make clean” i “make” per actualitzar els fitxers. A continuació executar el fitxer “execute.sh” mitjançant la comanda “./” i un cop realitzada la simulació executar el “post_process_plot_script.py”.

Un cop realitzat aquest procés, a la carpeta on es troben els fitxers estaran les gràfiques i el document amb els resultats “1d_high_pressure_framework_0.h5”. A continuació es mostra una captura de la finestra del putty.

```
jgorres@pc05:~$ cd work/flowsolverrhea/high_p/ld_nitrogen/pengrobinson/
jgorres@pc05:~/work/flowsolverrhea/high_p/ld_nitrogen/pengrobinson$ make clean
rm -f RHEA.exe
jgorres@pc05:~/work/flowsolverrhea/high_p/ld_nitrogen/pengrobinson$ make
mpic++ myRHEA.cpp -O3 -Wall -std=c++0x ../../../../src/*.cpp -o RHEA.exe -L/usr/li
b/x86_64-linux-gnu/hdf5/openmpi -I/usr/include/hdf5/openmpi -lyaml-cpp -lhdf5
jgorres@pc05:~/work/flowsolverrhea/high_p/ld_nitrogen/pengrobinson$ ./execute.sh

RHEA: START SIMULATION
Time advancement completed -> iteration = 0, time = 0.00000e+00 [s]
RHEA: END SIMULATION
jgorres@pc05:~/work/flowsolverrhea/high_p/ld_nitrogen/pengrobinson$ ./post_proce
ss_plot_script.py
Unable to init server: Could not connect: Connection refused
Unable to init server: No s'ha pogut connectar: Connection refused

(post_process_plot_script.py:111402): Gdk-CRITICAL **: 17:58:33.076: gdk_cursor_
new_for_display: assertion 'GDK_IS_DISPLAY (display)' failed
findfont: Font family ['sanserif'] not found. Falling back to DejaVu Sans.
23
1
jgorres@pc05:~/work/flowsolverrhea/high_p/ld_nitrogen/pengrobinson$
```

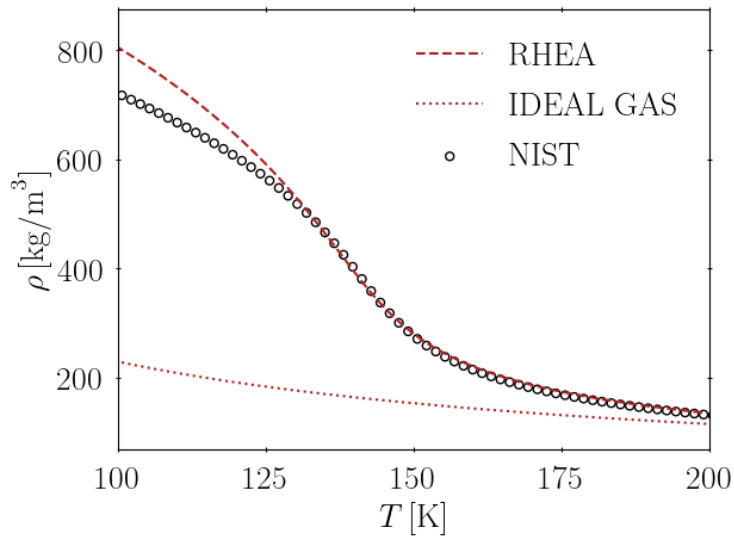
Figura 6-4. Execució fitxers (Font Josep Gorres, Elaboració pròpia)

7. Resultats

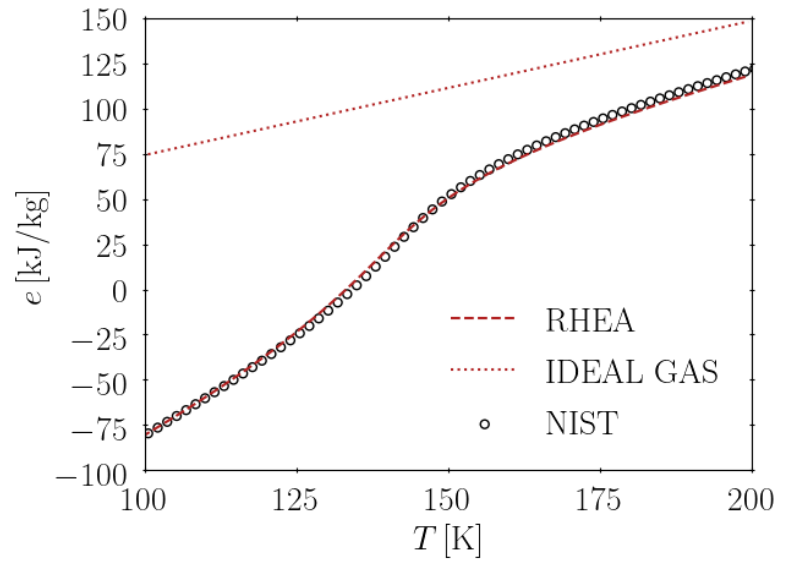
7.1. Nitrogen

7.1.1. 6,79 MPa

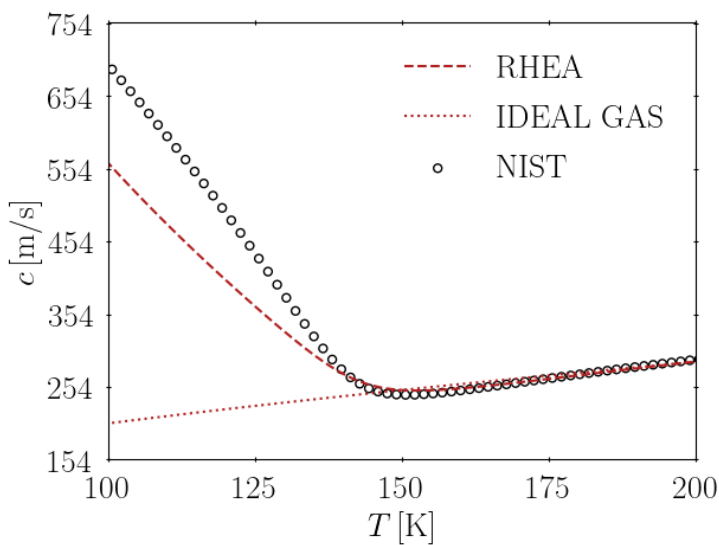
Density vs Temperature



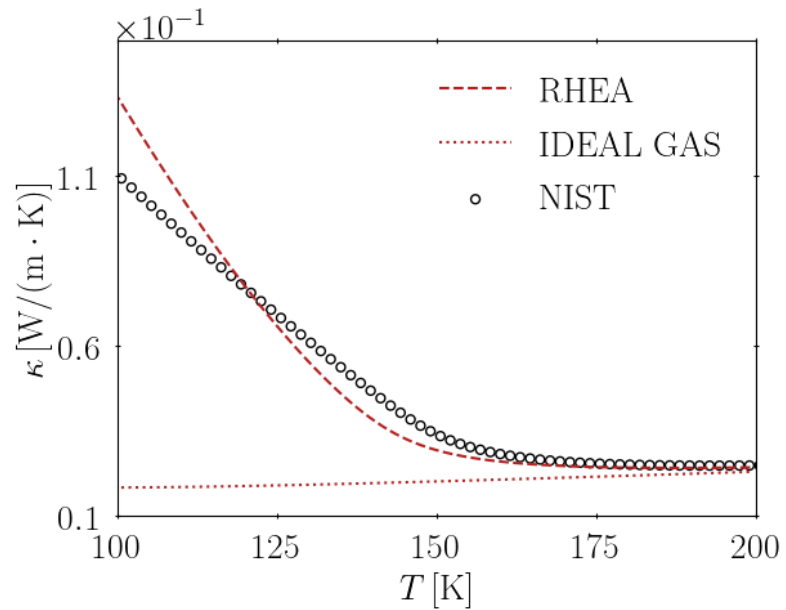
Internal Energy vs Temperature



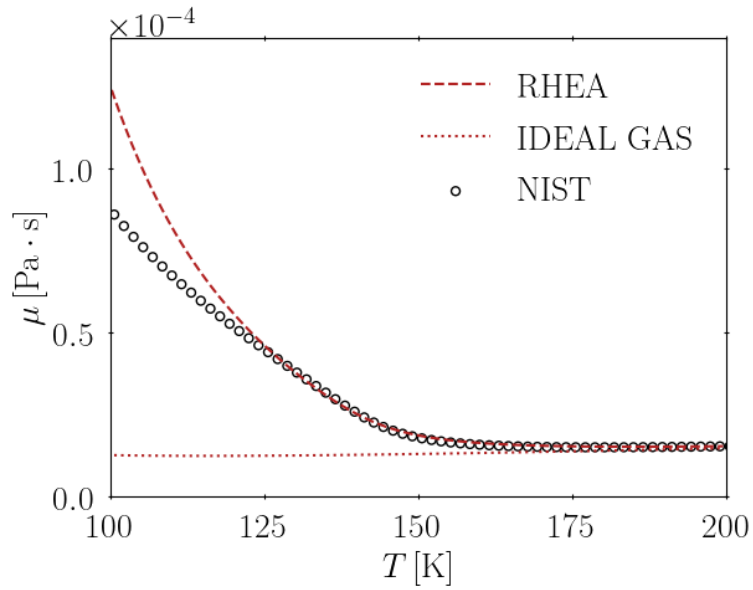
Sound speed vs Temperature



Thermal conductivity vs temperature

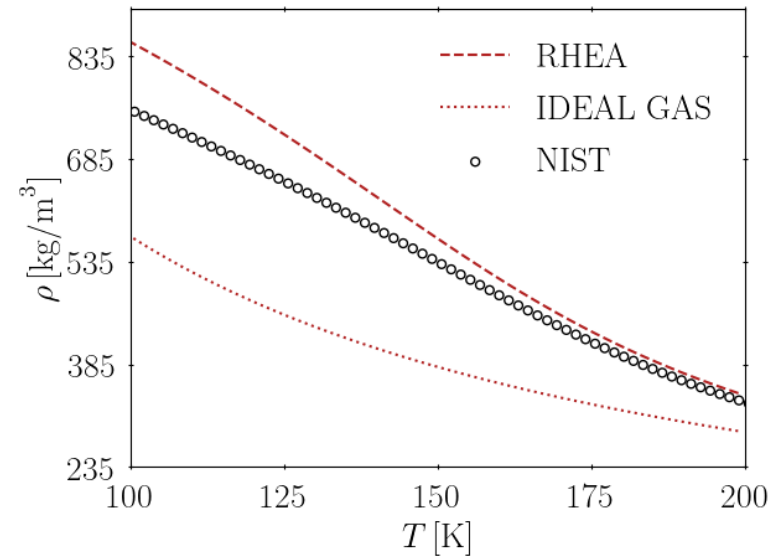


Viscosity vs Temperature

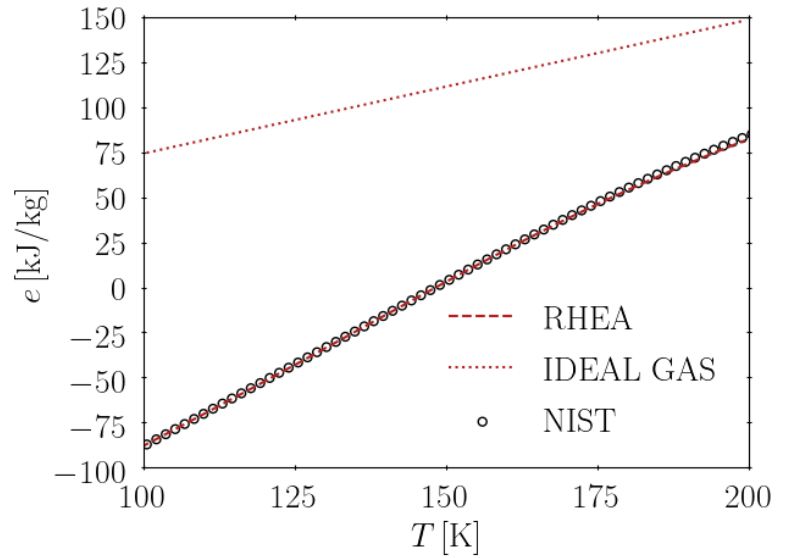


7.1.2. 16,97 Mpa

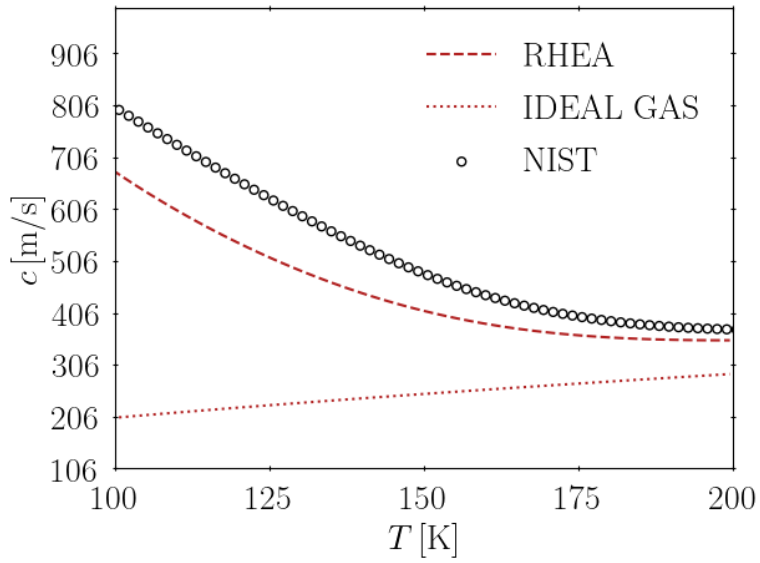
Density vs Temperature



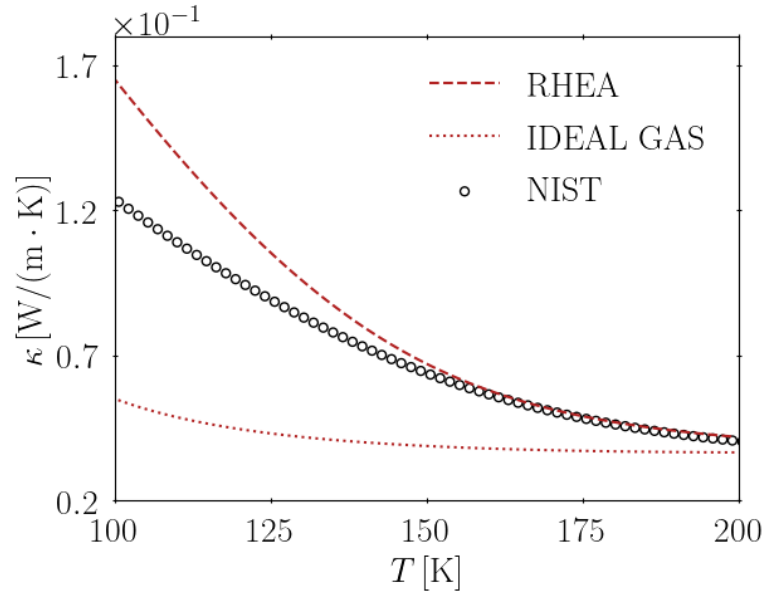
Internal Energy vs Temperature



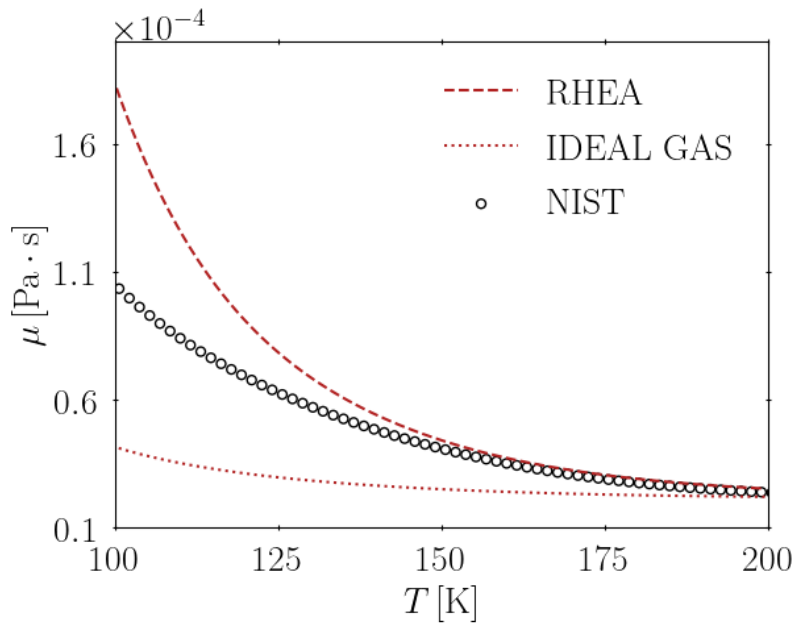
Sound speed vs Temperature



Thermal conductivity vs temperature

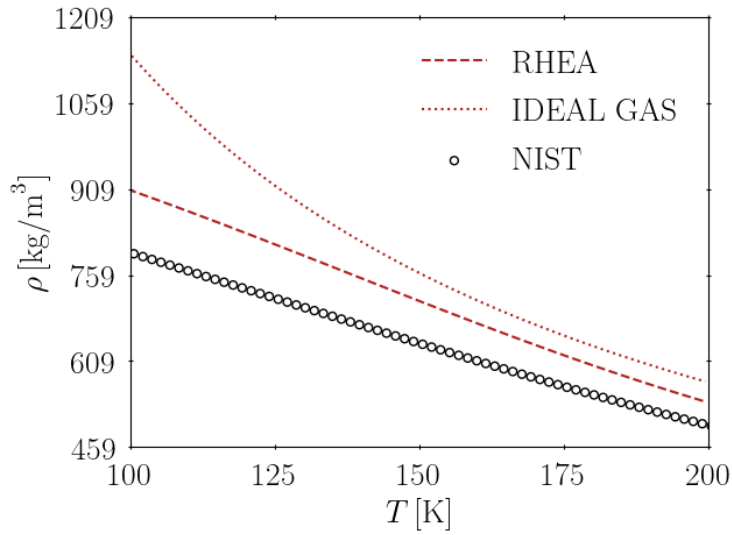


Viscosity vs Temperature

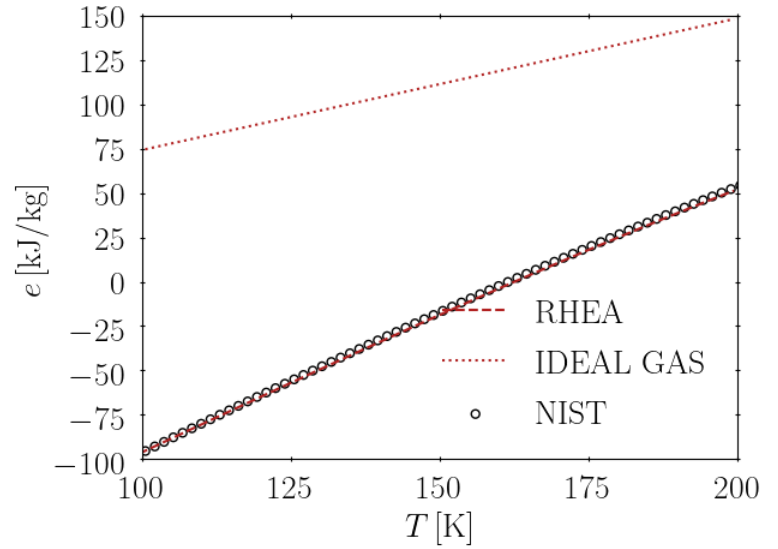


7.1.3. 33,95 MPa

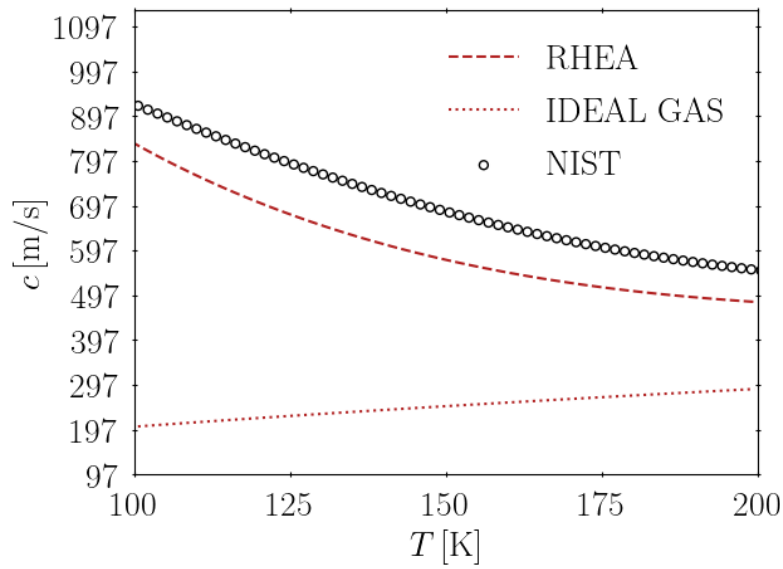
Density vs Temperature



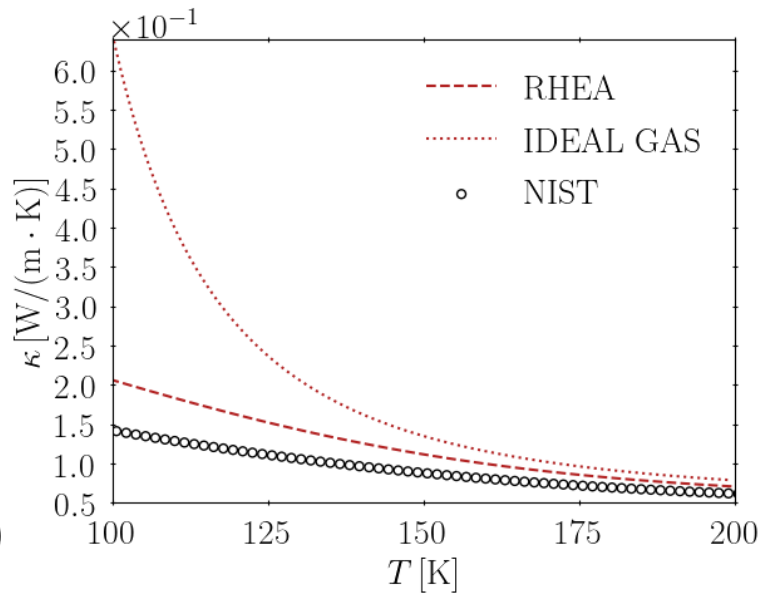
Internal Energy vs Temperature



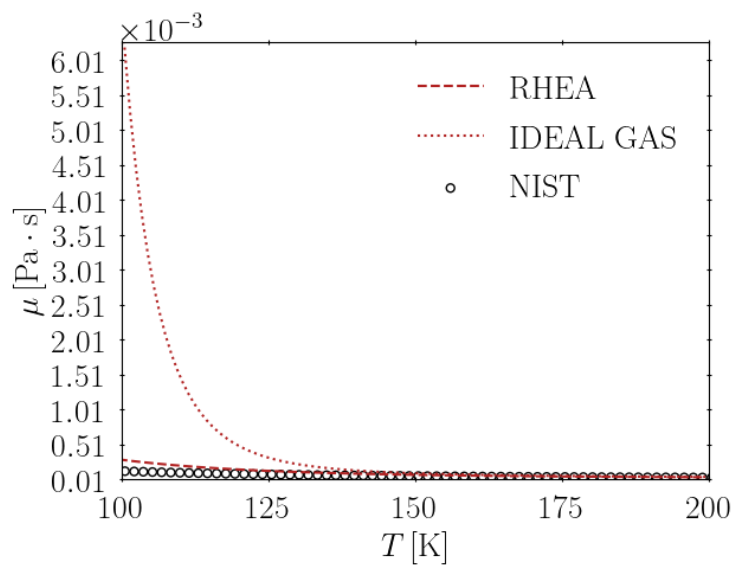
Sound speed vs Temperature



Thermal conductivity vs temperature

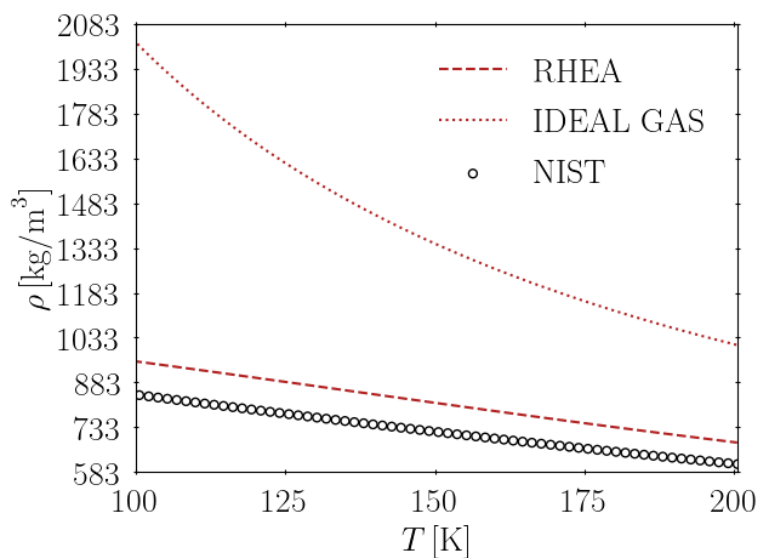


Viscosity vs Temperature

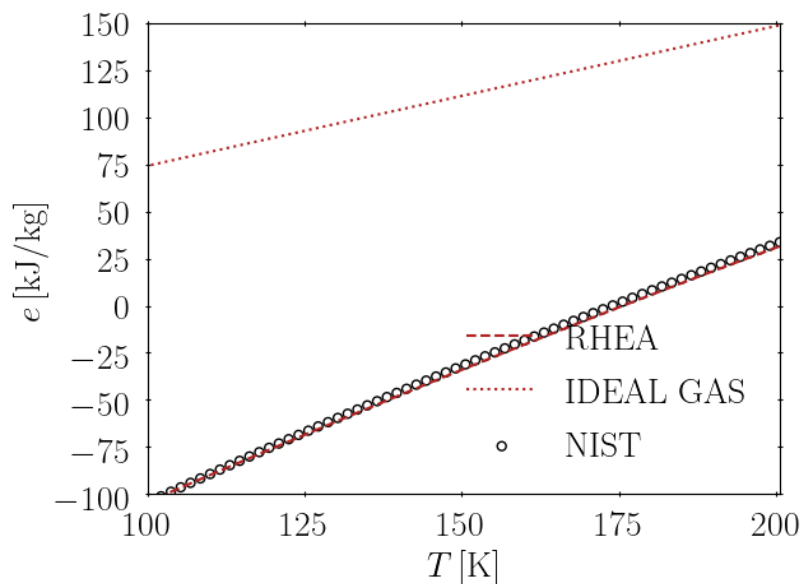


7.1.4. 60 MPa

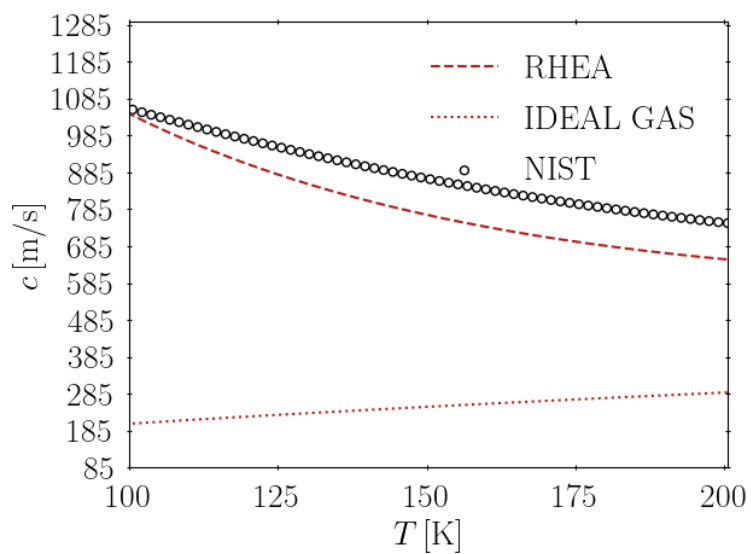
Density vs Temperature



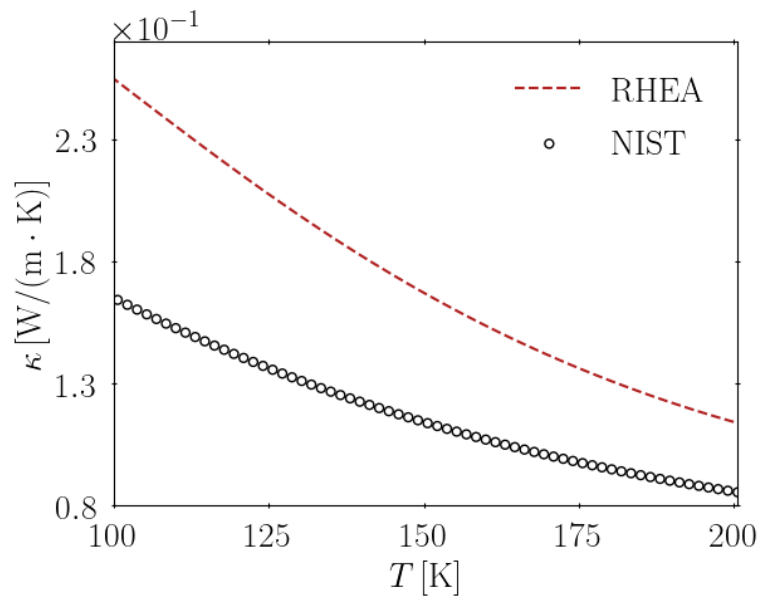
Internal Energy vs Temperature



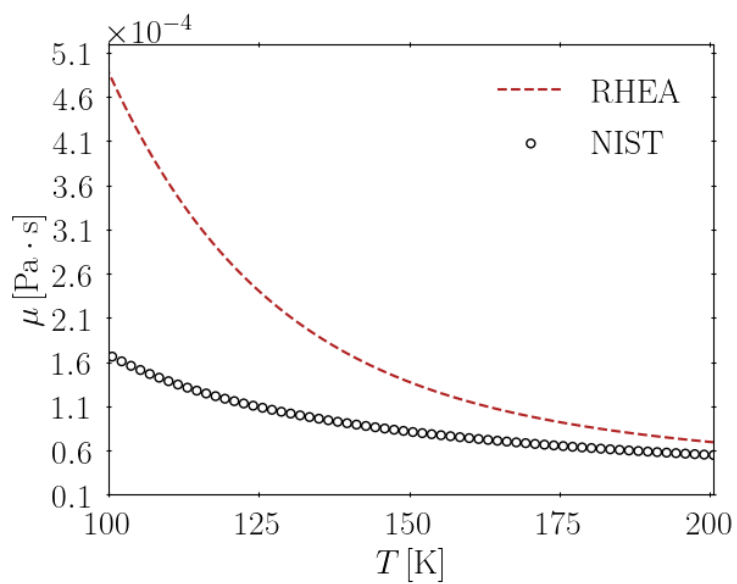
Sound speed vs Temperature



Thermal conductivity vs temperature

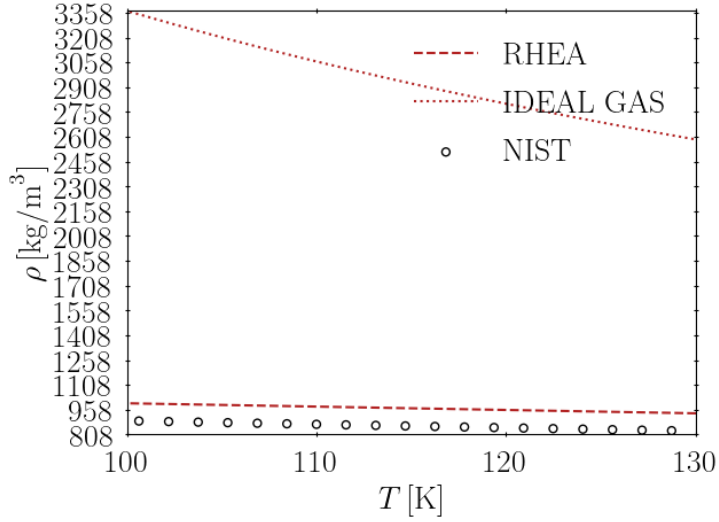


Viscosity vs Temperature

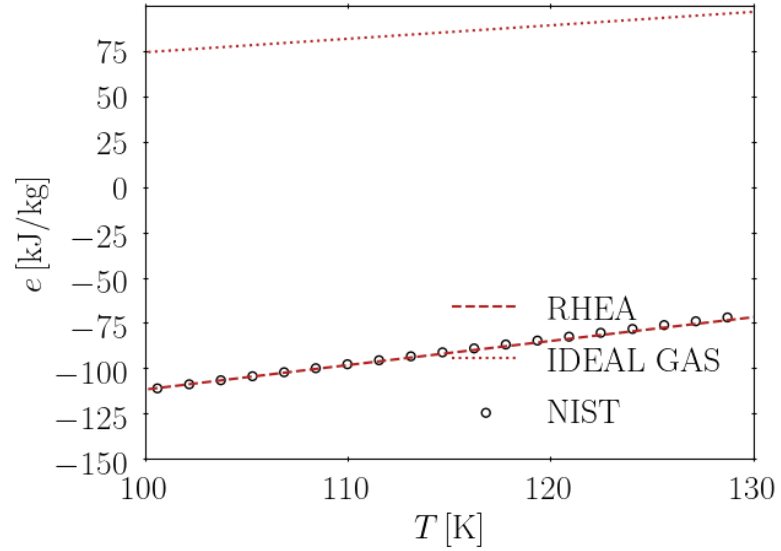


7.1.5. 100 MPa

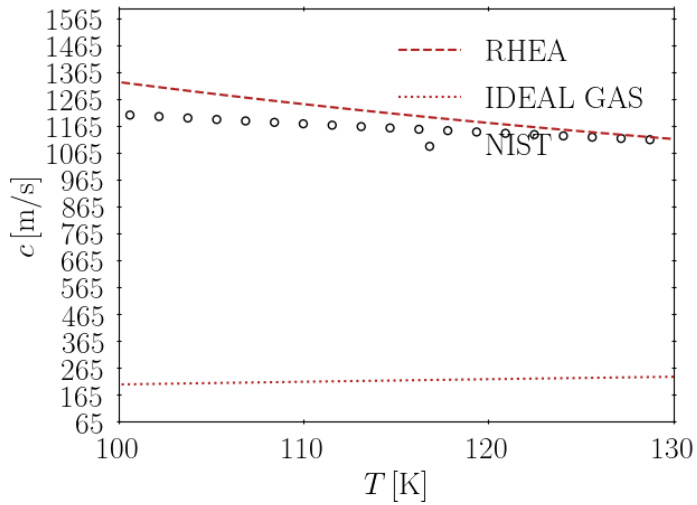
Density vs Temperature



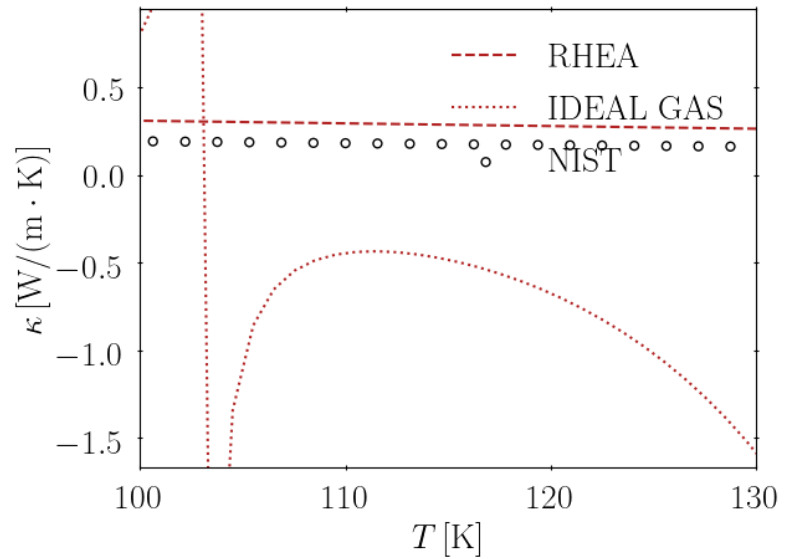
Internal Energy vs Temperature



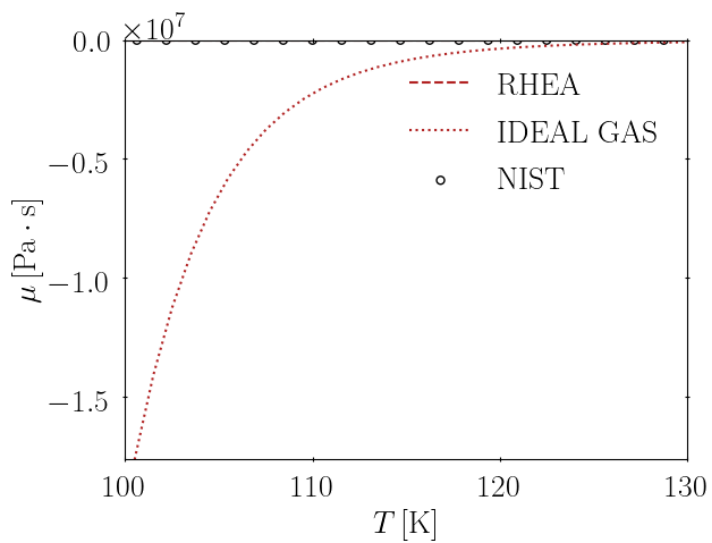
Sound speed vs Temperature



Thermal conductivity vs temperature





Viscosity vs Temperature




7.2. Taules resultats

Llegenda:

 Bona aproximació. La corba del flowsolverRHEA coincideix totalment amb la del NIST o majoritàriament.

 La corba del flowsolverRHEA coincideix en algun punt amb la del NIST o segueix la mateixa tendència, però amb un desfasament no molt significatiu.

 La corba del flowsolverRHEA no coincideix en cap punt amb la del NIST i està molt desfasada de la corba del NIST.

Substància: Nitrogen					
	Densitat	Energia Interna	Velocitat de so	Conductivitat Tèrmica	Viscositat
6,79 Mpa					
16,97 Mpa					
33,95 Mpa					
60 Mpa					
100 Mpa					

Taula 7-1. Resultats Nitrogen (Font: Josep Gorres, Elaboració pròpia)

Substància: Diòxid de Carboni					
	Densitat	Energia Interna	Velocitat de so	Conductivitat Tèrmica	Viscositat
14,75 Mpa					
36,89 Mpa					
73,77 Mpa					
140 Mpa					
200 Mpa					

Taula 7-2. Resultats Diòxid de Carboni (Font: Josep Gorres, Elaboració pròpia)

Substància: Dodecà					
	Densitat	Energia Interna	Velocitat de so	Conductivitat Tèrmica	Viscositat
3,63 Mpa					
9,09 Mpa					
18,17 Mpa					
50 Mpa					

Taula 7-3. Resultats Dodecà (Font: Josep Gorres, Elaboració pròpia)

Substància: Hidrogen					
	Densitat	Energia Interna	Velocitat de so	Conductivitat Tèrmica	Viscositat
2,59 Mpa					
6,48 Mpa					
12,96 Mpa					
20 Mpa					
30 Mpa					

Taula 7-4. Resultats Hidrogen (Font: Josep Gorres, Elaboració pròpia)

Substància: Oxigen					
	Densitat	Energia Interna	Velocitat de so	Conductivitat Tèrmica	Viscositat
10,09 Mpa					
25,22 Mpa					
50,43 Mpa					
65 Mpa					
82 Mpa					

Taula 7-5. Resultats Oxigen (Font: Josep Gorres, Elaboració pròpia)

Substància: Metà					
	Densitat	Energia Interna	Velocitat de so	Conductivitat Tèrmica	Viscositat
9,19 Mpa					
22,99 Mpa					
45,99 Mpa					
80 Mpa					
100 Mpa					

Taula 7-6. Resultats Metà (Font: Josep Gorres, Elaboració pròpia)

Anàlisi de impacte ambiental del projecte

Aquest projecte ha sigut purament un projecte de simulació i anàlisi numèric. Un programa de simulació computacional pot ajudar a prescindir de maquetes i de fabricar productes que desenvolupin bé la seva funció.

El fet de formar part del desenvolupament d'un programa de simulació computacional pot beneficiar en estalvis energètics. Un nou programa de simulació computacional més eficient pot influir a reduir els temps de computació i els seus consums energètics. Per una altra part, pot ajudar a estudiar millor els processos d'algunes màquines i reduir els seus consums energètics.

A pesar de les avantatges, l'únic impacte ambiental negatiu que se li pot considerar a aquest projecte, és el consum energètic dels ordinadors al llarg del projecte.

Conclusions

Durant el transcurs del present projecte s'ha validat el flowsolverRHEA, un model computacional capaç de realitzar simulacions de fluids en condicions supercrítiques utilitzant l'equació d'estat de Peng-Robinson. Per validar-lo, s'ha comparat amb les dades experimentals del NIST i l'equació d'estat de gas ideal.

Un cop s'han obtingut els primers resultats, s'ha pogut comprovar que l'equació d'estat de Peng-Robinson té limitacions. Per tal d'obtenir resultats fiables, s'ha estudiat en quins rangs de pressió i temperatura l'equació d'estat de Peng-Robinson funciona correctament. Després de veure els seus límits s'ha seguit realitzant simulacions ja en el rang que funciona l'equació d'estat de Peng-Robinson. Tot i les seves limitacions, el flowsolverRHEA pot arribar a realitzar càlculs fins a pressions elevades. En alguns casos com el del Nitrogen, Diòxid de Carboni i Dodecà de l'ordre de gairebé 30 vegades la pressió crítica de cada gas. El cas de l'oxigen només s'ha estudiat fins a 16 vegades la pressió crítica perquè al NIST no hi havia dades per pressions més elevades.

A [l'Annex B](#), es troben les gràfiques que s'obtenen del programa amb el Scilab. Aquestes gràfiques mostren fins quin temperatura funciona l'equació d'estat de Peng-Robinson a una determinada pressió. A mesura que augmenta la pressió, va disminuint la temperatura a la qual funciona l'equació d'estat de Peng-Robinson.

A [l'Annex C](#), es poden veure les gràfiques de totes les simulacions que s'han realitzat. Per cada substància es pot veure que a pressions no molt elevades, la corba del flowsolverRHEA segueix a la perfecció la corba del NIST. A mesura que es va augmentant la pressió, la corba del flowsolverRHEA es va distanciant de la del NIST, però tenint la mateixa forma. S'ha de tenir en compte que les dades del NIST tenen certa incertesa. No tenir la mateixa corba que el NIST, no significa que no sigui un bon resultat. Si es fa la comparació del flowsolverRHEA, que incorpora l'equació d'estat de Peng-Robinson, amb el model de gas ideal, s'observa que en aquestes simulacions el model de gas ideal no funciona. Les corbes resultat del model de gas ideal tenen valors desorbitats en alguns casos i la corba no s'assembla a la del NIST. Per tant, si es fa la comparació del flowsolverRHEA amb el model de gas ideal, el flowsolverRHEA obté un resultat molt més aproximat. Pot ser aquest resultat no serà el real, però ja és fiable.

Com es pot comprovar, s'han realitzat tots els objectius que s'havien proposat a l'inici del projecte. S'ha analitzat les equacions que incorpora i s'han realitzat el màxim de simulacions possibles amb el màxim de substàncies per validar-lo. Un altre objectiu que no està descrit, és executar un projecte que serveixi per fer servir el flowsolverRHEA des de zero. És a dir, que una persona que no l'ha utilitzat mai, seguint el projecte pugui arribar a saber fer-lo servir i entendre l'equació de Peng-Robinson

Bibliografia

- [1] Bahman Zohuri. Physics of Cryogenics, An Ultralow Temperature Phenomenon [En línia]. Albuquerque: Elsevier, 2018. ISBN 978-0-12-814519-7. Disponible a: <https://www.sciencedirect.com/book/9780128145197/physics-of-cryogenics?via=ihub>
- [2] Yunes A Çengel, Michael A.Boles. Termodinámica, 7ª edición. New York: McGraw Hill, 2011. ISBN 978-607-15-0743-3.
- [3] M. Salazar-Pereyra, Raúl Lugo Leyte, Juan M Zamora i O A Ruiz-Ramírez. Análisis Termodinámico De Los Ciclos Rankie y Subcríticos [En línia]. Oporto: 2011. Disponible a: https://www.researchgate.net/publication/319131137_ANALISIS_TERMODINAMICO_DE_LOS_CICLOS_RANKINE_SUPERCRITICOS_Y_SUBCRITICOS
- [4] DONALD B ROBINSON, DING-YU PENG i HENG-JOO NG. Applications of te Peng-Robinson Equation of State [En línia]. Canada: 1977. Disponible a: <https://pubs.acs.org/doi/abs/10.1021/bk-1977-0060.ch008>
- [5] Bruce E. Poling, John M. Prausnitz i John P. O'Connell. The properties of gases and líquids [En línia]. Disponible a: <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjrz9bo2YjxAhUB3xoKHUqDA1oQFjABegQIBBAD&url=https%3A%2F%2Fwww.researchgate.net%2Ffile.PostFileLoader.html%3Fid%3D576b60dc3d7f4b3e0b1acf84%26assetKey%3DAS%253A375993815584769%25401466655319665&usg=AOvVaw2K5A5cyGj1i7WXur3h2Mj>
- [6] Alexander Burcat, Branko Ruscic. Third Millennium Ideal Gas and Condensed Phase Thermochemical Database for Combustion with Updates from Active Thermochemical Tables. Chicago: 2005.
- [7] Libro del Web de Química del NIST. A NIST [En línia]. National Institute of Standarts and Technology [Consulta: 8 de juny de 2021]. Disponible a: <https://webbook.nist.gov/chemistry/>

Annex A: Documents

A1. Llibreria de substàncies

LIBRARY OF SUBSTANCES FILE (YAML LANGUAGE)

#

P. J. Linstrom and W. G. Mallard.

Chemistry WebBook, NIST Standard Reference Database.

<http://webbook.nist.gov/chemistry/fluid/>

#

A. Burcat and B. Ruscic

Third Millennium Ideal Gas and Condensed Phase Thermochemical Database

for Combustion with Updates from Active Thermochemical Tables.

ANL-05/20, TAE 960

NITROGEN

NITROGEN:

R_specific: 296.80 # Specific gas constant [J/(kg·K)]
gamma: 1.4 # Heat capacity ratio (ideal-gas) [-]
P_inf: 0.0 # Pressure infinity (liquid stiffness) [Pa]
e_0: 0.0 # Internal energy zero point [J/kg]
c_v: 0.0 # Specific isochoric heat capacity [J/(kg·K)]
molecular_weight: 0.0280134 # Molecular weight [kg/mol]
acentric_factor: 0.0372 # Acentric factor [-]
critical_temperature: 126.192 # Critical temperature [K]
critical_pressure: 3395800.0 # Critical pressure [Pa]
critical_molar_volume: 0.000089412 # Critical molar volume
[m3/mol]
NASA_coefficients: [2.9525763700000000000000,
0.0013969004000000000000,
-0.0000004926316030000000,
0.000000000078601019000,
-0.000000000000004607552,
-923.94868800000000000000,
5.8718876200000000000000,
3.5310052800000000000000,
-0.0001236609800000000000,
-0.0000005029994330000000,

```

0.000000002435306120000,
-0.000000000001408812400,
-1046.976280000000000000,
2.967470380000000000000,
0.00000000000000000000]          # NASA 7-coefficient polynomial (15 values)

dipole_moment: 0.0                  # Dipole moment [D]

association_factor: 0.0             # Association factor [-]

##### OXYGEN #####

OXYGEN:

R_specific: 259.82                  # Specific gas constant [J/(kg·K)]

gamma: 1.4                          # Heat capacity ratio (ideal-gas) [-]

P_inf: 0.0                          # Pressure infinity (liquid stiffness) [Pa]

e_0: 0.0                            # Internal energy zero point [J/kg]

c_v: 0.0                            # Specific isochoric heat capacity [J/(kg·K)]

molecular_weight: 0.031999         # Molecular weight [kg/mol]

acentric_factor: 0.0222            # Acentric factor [-]

critical_temperature: 154.58        # Critical temperature [K]

critical_pressure: 5043000.0        # Critical pressure [Pa]

critical_molar_volume: 0.000073375 # Critical molar volume
[m3/mol]

NASA_coefficients: [ 3.660960830000000000000000,
0.000656365523000000000000,
-0.00000014114948500000000,

```

```

0.00000000002057976580000,
-0.00000000000000129913248,
-1215.97725000000000000000,
3.415361840000000000000000,
3.782456360000000000000000,
-0.002996734150000000000000,
0.000009847302000000000000,
-0.000000009681295080000000,
0.0000000000324372836000,
-1063.9435600000000000000000,
3.657675730000000000000000,
0.000000000000000000000000] # NASA 7-coefficient polynomial (15 values)

```

dipole_moment: 0.0

Dipole moment [D]

association_factor: 0.0

Association factor [-]

CARBON DIOXIDE

CARBON_DIOXIDE:

R_specific: 188.91

Specific gas constant [J/(kg·K)]

gamma: 1.29

Heat capacity ratio (ideal-gas) [-]

P_inf: 0.0

Pressure infinity (liquid stiffness) [Pa]

e_0: 0.0

Internal energy zero point [J/kg]

c_v: 0.0

Specific isochoric heat capacity [J/(kg·K)]

molecular_weight: 0.0440098

Molecular weight [kg/mol]

acentric_factor: 0.2239 # Acentric factor [-]

critical_temperature: 304.128 # Critical temperature [K]

critical_pressure: 7377300.0 # Critical pressure [Pa]

critical_molar_volume: 0.000094118 # Critical molar volume
[m3/mol]

NASA_coefficients: [4.636511100000000000000000,
0.002741456900000000000000,
-0.000000995897590000000000,
0.0000000001603866600000,
-0.00000000000000091619857,
-49024.904000000000000000,
-1.9348955000000000000000,
2.3568130000000000000000,
0.008984129900000000000000,
-0.000007122063200000000000,
0.000000002457300800000000,
-0.0000000000001428854800,
-48371.971000000000000000,
9.9009035000000000000000,
-47328.105000000000000000]

NASA 7-coefficient polynomial (15 values)

dipole_moment: 0.0 # Dipole moment [D]

association_factor: 0.0 # Association factor [-]

HYDROGEN

HYDROGEN:

R_specific: 4124.25 # Specific gas constant [J/(kg·K)]
 gamma: 1.40 # Heat capacity ratio (ideal-gas) [-]
 P_inf: 0.0 # Pressure infinity (liquid stiffness) [Pa]
 e_0: 0.0 # Internal energy zero point [J/kg]
 c_v: 0.0 # Specific isochoric heat capacity [J/(kg·K)]
 molecular_weight: 0.0020159 # Molecular weight [kg/mol]
 acentric_factor: -0.219 # Acentric factor [-]
 critical_temperature: 33.145 # Critical temperature [K]
 critical_pressure: 1296400.0 # Critical pressure [Pa]
 critical_molar_volume: 0.000064481 # Critical molar volume
 [m3/mol]
 NASA_coefficients: [2.932830500000000000000000,
 0.000826598020000000000000,
 -0.000000146400570000000000,
 0.000000000015409851000000,
 -0.000000000000000068879615,
 -813.055820000000000000000000,
 -1.02431640000000000000000000,
 2.34430290000000000000000000,
 0.00798042480000000000000000,
 -0.000019477917000000000000,


```

0.00000002015696700000000,
-0.000000000000737602890000,
-917.9241300000000000000000,
0.683002180000000000000000,
0.000000000000000000000000] # NASA 7-coefficient polynomial (15 values)

dipole_moment: 0.0 # Dipole moment [D]

association_factor: 0.0 # Association factor [-]

##### METHANE_RRHO #####

METHANE_RRHO:

R_specific: 518.24 # Specific gas constant [J/(kg·K)]

gamma: 1.30 # Heat capacity ratio (ideal-gas) [-]

P_inf: 0.0 # Pressure infinity (liquid stiffness) [Pa]

e_0: 0.0 # Internal energy zero point [J/kg]

c_v: 0.0 # Specific isochoric heat capacity [J/(kg·K)]

molecular_weight: 0.016043 # Molecular weight [kg/mol]

acentric_factor: 0.01142 # Acentric factor [-]

critical_temperature: 190.564 # Critical temperature [K]

critical_pressure: 4599200.0 # Critical pressure [Pa]

critical_molar_volume: 0.000098628 # Critical molar volume
[m3/mol]

NASA_coefficients: [ 1.9117860000000000000000,

0.0096026796000000000000,

-0.0000033838784100000000,

```

```

0.0000000005387972400000,
-0.00000000000000319306807,
-10099.213600000000000000,
8.4824186100000000000000,
5.1482573200000000000000,
-0.0137002410000000000000,
0.0000493749414000000000,
-0.0000000491952339000000,
0.000000000170097299000,
-10245.322200000000000000,
-4.6332272600000000000000,
-8972.2665600000000000000] # NASA 7-coefficient polynomial (15 values)

```

dipole_moment: 0.0 # Dipole moment [D]

association_factor: 0.0 # Association factor [-]

DODECANE

DODECANE:

R_specific: 47.75 # Specific gas constant [J/(kg·K)]

gamma: 1.03 # Heat capacity ratio (ideal-gas) [-]

P_inf: 0.0 # Pressure infinity (liquid stiffness) [Pa]

e_0: 0.0 # Internal energy zero point [J/kg]

c_v: 0.0 # Specific isochoric heat capacity [J/(kg·K)]

molecular_weight: 0.170338 # Molecular weight [kg/mol]

acentric_factor: 0.574 # Acentric factor [-]

critical_temperature: 658.1 # Critical temperature [K]

critical_pressure: 1817000.0 # Critical pressure [Pa]

critical_molar_volume: 0.000750389 # Critical molar volume
[m3/mol]

NASA_coefficients: [37.0187925000000000000000,
0.0554721488000000000000,
-0.0000192079548000000000,
0.000000003081755740000,
-0.000000000000184800617,
-52698.445800000000000000,
-161.45350100000000000000,
21.3264480000000000000000,
-0.0386394002000000000000,
0.0003994761130000000000,
-0.000000506681097000000,
0.000000000200697878000,
-42247.505300000000000000,
-48.5848300000000000000000,
-34983.622600000000000000]

NASA 7-coefficient polynomial (15 values)

dipole_moment: 0.0 # Dipole moment [D]

association_factor: 0.0 # Association factor [-]

A2. Substances_selection

```
if( substance_name == "NITROGEN" ) {  
    substance = substances_library["NITROGEN"];  
} else if ( substance_name == "OXYGEN" ) {  
    substance = substances_library["OXYGEN"];  
} else if ( substance_name == "HYDROGEN" ) {  
    substance = substances_library["HYDROGEN"];  
} else if ( substance_name == "CARBON_DIOXIDE" ) {  
    substance = substances_library["CARBON_DIOXIDE"];  
} else if ( substance_name == "METHANE_RRHO" ) {  
    substance = substances_library["METHANE_RRHO"];  
} else if ( substance_name == "DODECANE" ) {  
    substance = substances_library["DODECANE"];  
} else {  
    cout << "Substance not available!" << endl;  
    MPI_Abort( MPI_COMM_WORLD, 1 );  
};
```

A3. Postprocessat original

```
#!/usr/bin/python3

import sys
import os
import numpy as np
import h5py
import matplotlib.pyplot as plt
from matplotlib.transforms import Bbox
from matplotlib import rc,rcParams
from scipy.optimize import fsolve
plt.rc( 'text', usetex = True )
rc('font', family='sanserif')
plt.rc( 'font', size = 20 )
plt.rcParams['text.latex.preamble'] = [ r'\usepackage{amsmath}', r'\usepackage{amssymb}', r'\usepackage{color}'
]

### Open data file
data_file = h5py.File( '1d_high_pressure_framework_0.h5', 'r' )
#list( data_file.keys() )
x_data = data_file['x'][0,0,:]; x_data = np.asarray( x_data.flatten() )
y_data = data_file['y'][0,0,:]; y_data = np.asarray( y_data.flatten() )
z_data = data_file['z'][0,0,:]; z_data = np.asarray( z_data.flatten() )
rho_data = data_file['rho'][0,0,:]; rho_data = np.asarray( rho_data.flatten() )
u_data = data_file['u'][0,0,:]; u_data = np.asarray( u_data.flatten() )
v_data = data_file['v'][0,0,:]; v_data = np.asarray( v_data.flatten() )
w_data = data_file['w'][0,0,:]; w_data = np.asarray( w_data.flatten() )
E_data = data_file['E'][0,0,:]; E_data = np.asarray( E_data.flatten() )
P_data = data_file['P'][0,0,:]; P_data = np.asarray( P_data.flatten() )
T_data = data_file['T'][0,0,:]; T_data = np.asarray( T_data.flatten() )
sos_data = data_file['sos'][0,0,:]; sos_data = np.asarray( sos_data.flatten() )
mu_data = data_file['mu'][0,0,:]; mu_data = np.asarray( mu_data.flatten() )
kappa_data = data_file['kappa'][0,0,:]; kappa_data = np.asarray( kappa_data.flatten() )
ke_data = 0.5*( u_data*u_data + v_data*v_data + w_data*w_data )
e_data = E_data - ke_data

### Open NIST file
T_nist, P_nist, rho_nist, V_nist, e_nist, h_nist, s_nist, cv_nist, cp_nist, sos_nist, JT_nist, mu_nist, kappa_nist =
np.loadtxt( 'nist_nitrogen_4MPa_0-1000K.csv', delimiter=',', unpack = 'True' )

### Plot density vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, rho_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'$\text{NIST}$' )
plt.plot( T_data, rho_data, linestyle = '--', color = 'firebrick', label = r'$\text{RHEA}$' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = 'True', direction = 'inout')
plt.xlim( 100, 200 )
plt.xticks( np.arange( 100, 210, 25 ) )
plt.xscale( 'log' )
plt.xlabel( r'$T \text{ \thinspace } \text{K}$' )
plt.ylim( 50, 800 )
plt.yticks( np.arange( 50, 810, 150 ) )
plt.yscale( 'log' )
plt.ylabel( r'$\rho \text{ \thinspace } \text{kg/m}^3$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
```

```

plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'density_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )

#### Plot internal energy vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, e_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\textrm{NIST}$' )
plt.plot( T_data, e_data/1.0e3 + ( 387.676 - 76.809 ), linestyle = '--', color = 'firebrick', label = r'\textrm{RHEA}$' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.xlim( 100, 200 )
plt.xticks( np.arange( 100, 210, 25 ) )
#plt.xscale( 'log' )
plt.xlabel( r'$T \thinspace \textrm{[K]}$' )
plt.ylim( -100, 150 )
plt.yticks( np.arange( -100, 160, 50 ) )
#plt.yscale( 'log' )
plt.ylabel( r'$e \thinspace \textrm{[kJ/kg]}$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper left' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'internal_energy_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )

#### Plot speed of sound vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, sos_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\textrm{NIST}$' )
plt.plot( T_data, sos_data, linestyle = '--', color = 'firebrick', label = r'\textrm{RHEA}$' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.xlim( 100, 200 )
plt.xticks( np.arange( 100, 210, 25 ) )
#plt.xscale( 'log' )
plt.xlabel( r'$T \thinspace \textrm{[K]}$' )
plt.ylim( 100, 700 )
plt.yticks( np.arange( 100, 710, 100 ) )
#plt.yscale( 'log' )
plt.ylabel( r'$c \thinspace \textrm{[m/s]}$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'sound_speed_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )

#### Plot viscosity vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, mu_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\textrm{NIST}$' )
plt.plot( T_data, mu_data, linestyle = '--', color = 'firebrick', label = r'\textrm{RHEA}$' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.ticklabel_format(axis = 'y', style = 'sci', scilimits=(0,0))
plt.xlim( 100, 200 )

```

```

plt.xticks( np.arange( 100, 210, 25 ) )
#plt.xscale( 'log' )
plt.xlabel( r'$T \text{ \thinspace } \text{\texttrm}[K]$\text{' } )
plt.ylim( 0.0, 0.00015 )
plt.yticks( np.arange( 0.0, 0.00016, 0.00005 ) )
#plt.yscale( 'log' )
plt.ylabel( r'$\mu \text{ \thinspace } \text{\texttrm}[Pa] \cdot \text{\texttrm}[s]$\text{' } )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'viscosity_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )

### Plot thermal conductivity vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, kappa_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'$\text{\texttrm}[NIST]$\text{' } )
plt.plot( T_data, kappa_data, linestyle = '--', color = 'firebrick', label = r'$\text{\texttrm}[RHEA]$\text{' } )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.ticklabel_format(axis = 'y', style = 'sci', scilimits=(0,0))
plt.xlim( 100, 200 )
plt.xticks( np.arange( 100, 210, 25 ) )
#plt.xscale( 'log' )
plt.xlabel( r'$T \text{ \thinspace } \text{\texttrm}[K]$\text{' } )
plt.ylim( 0.0, 0.15 )
plt.yticks( np.arange( 0.0, 0.16, 0.05 ) )
#plt.yscale( 'log' )
plt.ylabel( r'$\kappa \text{ \thinspace } \text{\texttrm}[W/(m) \cdot \text{\texttrm}[K]]$\text{' } )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'thermal_conductivity_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )

```

A4. Postprocessat NIST, RHEA

```
#!/usr/bin/python3

import sys
import os
import numpy as np
import h5py
import matplotlib.pyplot as plt
from matplotlib.transforms import Bbox
from matplotlib import rc,rcParams
from scipy.optimize import fsolve
plt.rc('text', usetex = True )
rc('font', family='sanserif')
plt.rc('font', size = 20 )
plt.rcParams['text.latex.preamble'] = [ r'\usepackage{amsmath}', r'\usepackage{amssymb}', r'\usepackage{color}'
]

### Open data file
data_file = h5py.File('1d_high_pressure_framework_0.h5', 'r')
#list( data_file.keys() )
x_data = data_file['x'][0,0,:]; x_data = np.asarray( x_data.flatten() )
y_data = data_file['y'][0,0,:]; y_data = np.asarray( y_data.flatten() )
z_data = data_file['z'][0,0,:]; z_data = np.asarray( z_data.flatten() )
rho_data = data_file['rho'][0,0,:]; rho_data = np.asarray( rho_data.flatten() )
u_data = data_file['u'][0,0,:]; u_data = np.asarray( u_data.flatten() )
v_data = data_file['v'][0,0,:]; v_data = np.asarray( v_data.flatten() )
w_data = data_file['w'][0,0,:]; w_data = np.asarray( w_data.flatten() )
E_data = data_file['E'][0,0,:]; E_data = np.asarray( E_data.flatten() )
P_data = data_file['P'][0,0,:]; P_data = np.asarray( P_data.flatten() )
T_data = data_file['T'][0,0,:]; T_data = np.asarray( T_data.flatten() )
sos_data = data_file['sos'][0,0,:]; sos_data = np.asarray( sos_data.flatten() )
mu_data = data_file['mu'][0,0,:]; mu_data = np.asarray( mu_data.flatten() )
kappa_data = data_file['kappa'][0,0,:]; kappa_data = np.asarray( kappa_data.flatten() )
ke_data = 0.5*( u_data*u_data + v_data*v_data + w_data*w_data )
e_data = E_data - ke_data

### Open NIST file
T_nist, P_nist, rho_nist, V_nist, e_nist, h_nist, s_nist, cv_nist, cp_nist, sos_nist, JT_nist, mu_nist, kappa_nist =
np.loadtxt('oxygen_82MPa_0-1000K.csv', delimiter=',', unpack = 'True' )

t_min = round(T_data[0], 0) # Temperature [K]
t_max = T_data[-1] # Temperature [K]

posN1 = 0
while T_nist[posN1] < t_min:
    posN1 += 1

print(posN1)

posN2 = 0
while T_nist[posN2] < t_max:
    posN2 += 1

print(posN2)

posD1 = 0
while T_data[posD1] < t_min:
    posD1 += 1

print(posD1)
```



```

posD2 = 0
while T_data[posD2] < t_max:
    posD2 += 1

print(posD2)

x= (rho_data[posD1], rho_data[posD2], rho_nist[posN1], rho_nist[posN2])
rho_min= round(min(x)-0.1*abs(max(rho_data)-min(rho_data)), 0)
rho_max= round(max(x)+0.1*abs(max(rho_data)-min(rho_data)), 0)
sos_min= round(min(sos_data)-0.1*(max(sos_data)-min(sos_data)), 0)
sos_max= round(max(sos_data)+0.1*(max(sos_data)-min(sos_data)), 0)
y= (mu_data[posD1], mu_data[posD2], mu_nist[posN1], mu_nist[posN2])
mu_min= round(min(y)-0.1*(max(mu_data)-min(mu_data)), 5)
mu_max= round(max(y)+0.1*(max(mu_data)-min(mu_data)), 5)
z= (kappa_data[posD1], kappa_data[posD2], kappa_nist[posN1], kappa_nist[posN2])
kappa_min= round(min(z)-0.1*(min(z)), 2)
kappa_max= round(max(z)+0.1*(max(kappa_data)-min(kappa_data)), 2)
print(z)
print(kappa_min)
print(kappa_max)

### Plot density vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, rho_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\textm{NIST}' )
plt.plot( T_data, rho_data, linestyle = '--', color = 'firebrick', label = r'\textm{RHEA}' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = 'True', direction = 'inout')
plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 25 ) )
#plt.xscale( 'log' )
plt.xlabel( r'$T \text{ \thinspace } \text{m}[K] $' )
plt.ylim( rho_min, rho_max )
plt.yticks( np.arange( rho_min, rho_max + 50, 100 ) )
#plt.yscale( 'log' )
plt.ylabel( r'$\rho \text{ \thinspace } \text{m}[kg/m]^3 \text{m} $' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'density_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'density_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

### Plot internal energy vs. temperature

posN = 0
while T_nist[posN] < t_min:
    posN += 1

print(posN)

posD = 0
while T_data[posD] < t_min:
    posD += 1

print(posD)

offsetE=e_nist[posN]-e_data[posD]/1.0e3;

```



```

E_min= round(E_data[0]/1.0e3+offsetE-abs(0.1*(abs(E_data[-1]/1.0e3)-abs(E_data[0]/1.0e3))), 0)
E_max= round(E_data[-1]/1.0e3+offsetE+abs(0.1*(abs(E_data[-1]/1.0e3)-abs(E_data[0]/1.0e3))), 0)

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, e_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\textrm{NIST}$' )
plt.plot( T_data, e_data/1.0e3 + offsetE, linestyle = '--', color = 'firebrick', label = r'\textrm{RHEA}$' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 25 ) )
plt.xscale( 'log' )
plt.xlabel( r'$T \thinspace \textrm{[K]}$' )
plt.ylim( E_min, E_max )
plt.yticks( np.arange( E_min, E_max+ 10, 20 ) )
plt.yscale( 'log' )
plt.ylabel( r'$e \thinspace \textrm{[kJ/kg]}$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper left' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'internal_energy_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'internal_energy_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

#### Plot speed of sound vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, sos_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\textrm{NIST}$' )
plt.plot( T_data, sos_data, linestyle = '--', color = 'firebrick', label = r'\textrm{RHEA}$' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 25 ) )
plt.xscale( 'log' )
plt.xlabel( r'$T \thinspace \textrm{[K]}$' )
plt.ylim( sos_min, sos_max )
plt.yticks( np.arange( sos_min, sos_max + 10, 50 ) )
plt.yscale( 'log' )
plt.ylabel( r'$c \thinspace \textrm{[m/s]}$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'sound_speed_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'sound_speed_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

#### Plot viscosity vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, mu_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\textrm{NIST}$' )
plt.plot( T_data, mu_data, linestyle = '--', color = 'firebrick', label = r'\textrm{RHEA}$' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.ticklabel_format(axis = 'y', style = 'sci', scilimits=(0,0))

```

```

plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 25 ) )
plt.xscale( 'log' )
plt.xlabel( r'$T \text{ \thinspace } \text{\texttrm{[K]}}$' )
plt.ylim( mu_min, mu_max )
plt.yticks( np.arange( mu_min, mu_max + 0.00001, 0.00005 ) )
plt.yscale( 'log' )
plt.ylabel( r'$\mu \text{ \thinspace } \text{\texttrm{[Pa]}} \cdot \text{\texttrm{[s]}}$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'viscosity_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'viscosity_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

### Plot thermal conductivity vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, kappa_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'$\text{\texttrm{NIST}}$' )
plt.plot( T_data, kappa_data, linestyle = '--', color = 'firebrick', label = r'$\text{\texttrm{RHEA}}$' )

# Configure plot
plt.tick_params( axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout' )
plt.ticklabel_format( axis = 'y', style = 'sci', scilimits=(0,0) )
plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 25 ) )
plt.xscale( 'log' )
plt.xlabel( r'$T \text{ \thinspace } \text{\texttrm{[K]}}$' )
plt.ylim( kappa_min, kappa_max )
plt.yticks( np.arange( kappa_min, kappa_max + 0.01, 0.05 ) )
plt.yscale( 'log' )
plt.ylabel( r'$\kappa \text{ \thinspace } \text{\texttrm{[W/(m)}} \cdot \text{\texttrm{[K]}}$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'thermal_conductivity_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'thermal_conductivity_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

```

A5. Postprocessat NIST, RHEA i IDEAL GAS

```
#!/usr/bin/python3

import sys
import os
import numpy as np
import h5py
import matplotlib.pyplot as plt
from matplotlib.transforms import Bbox
from matplotlib import rc,rcParams
from scipy.optimize import fsolve
plt.rc('text', usetex = True )
rc('font', family='sanserif')
plt.rc('font', size = 20 )
plt.rcParams['text.latex.preamble'] = [ r'\usepackage{amsmath}', r'\usepackage{amssymb}', r'\usepackage{color}'
]

### Open data file
data_file = h5py.File('1d_high_pressure_framework_0.h5', 'r')
#list( data_file.keys() )
x_data = data_file['x'][0,0,:]; x_data = np.asarray( x_data.flatten() )
y_data = data_file['y'][0,0,:]; y_data = np.asarray( y_data.flatten() )
z_data = data_file['z'][0,0,:]; z_data = np.asarray( z_data.flatten() )
rho_data = data_file['rho'][0,0,:]; rho_data = np.asarray( rho_data.flatten() )
u_data = data_file['u'][0,0,:]; u_data = np.asarray( u_data.flatten() )
v_data = data_file['v'][0,0,:]; v_data = np.asarray( v_data.flatten() )
w_data = data_file['w'][0,0,:]; w_data = np.asarray( w_data.flatten() )
E_data = data_file['E'][0,0,:]; E_data = np.asarray( E_data.flatten() )
P_data = data_file['P'][0,0,:]; P_data = np.asarray( P_data.flatten() )
T_data = data_file['T'][0,0,:]; T_data = np.asarray( T_data.flatten() )
sos_data = data_file['sos'][0,0,:]; sos_data = np.asarray( sos_data.flatten() )
mu_data = data_file['mu'][0,0,:]; mu_data = np.asarray( mu_data.flatten() )
kappa_data = data_file['kappa'][0,0,:]; kappa_data = np.asarray( kappa_data.flatten() )
ke_data = 0.5*( u_data*u_data + v_data*v_data + w_data*w_data )
e_data = E_data - ke_data

### Open data file
data_file = h5py.File('1d_high_pressure_framework_ig_0.h5', 'r')
#list( data_file.keys() )
x_dataig = data_file['x'][0,0,:]; x_dataig = np.asarray( x_dataig.flatten() )
y_dataig = data_file['y'][0,0,:]; y_dataig = np.asarray( y_dataig.flatten() )
z_dataig = data_file['z'][0,0,:]; z_dataig = np.asarray( z_dataig.flatten() )
rho_dataig = data_file['rho'][0,0,:]; rho_dataig = np.asarray( rho_dataig.flatten() )
u_dataig = data_file['u'][0,0,:]; u_dataig = np.asarray( u_dataig.flatten() )
v_dataig = data_file['v'][0,0,:]; v_dataig = np.asarray( v_dataig.flatten() )
w_dataig = data_file['w'][0,0,:]; w_dataig = np.asarray( w_dataig.flatten() )
E_dataig = data_file['E'][0,0,:]; E_dataig = np.asarray( E_dataig.flatten() )
P_dataig = data_file['P'][0,0,:]; P_dataig = np.asarray( P_dataig.flatten() )
T_dataig = data_file['T'][0,0,:]; T_dataig = np.asarray( T_dataig.flatten() )
sos_dataig = data_file['sos'][0,0,:]; sos_dataig = np.asarray( sos_dataig.flatten() )
mu_dataig = data_file['mu'][0,0,:]; mu_dataig = np.asarray( mu_dataig.flatten() )
kappa_dataig = data_file['kappa'][0,0,:]; kappa_dataig = np.asarray( kappa_dataig.flatten() )
ke_dataig = 0.5*( u_dataig*u_dataig + v_dataig*v_dataig + w_dataig*w_dataig )
e_dataig = E_dataig - ke_dataig

### Open NIST file
T_nist, P_nist, rho_nist, V_nist, e_nist, h_nist, s_nist, cv_nist, cp_nist, JT_nist, mu_nist, kappa_nist =
np.loadtxt('oxygen_82MPa_0-1000K.csv', delimiter=',', unpack = 'True')

t_min = round(T_data[0], 0) # Temperature [K]
```

```

t_max = T_data[-1]      # Temperature [K]

posN1 = 0
while T_nist[posN1] < t_min:
    posN1 += 1

print(posN1)

posN2 = 0
while T_nist[posN2] < t_max:
    posN2 += 1

print(posN2)

posD1 = 0
while T_data[posD1] < t_min:
    posD1 += 1

print(posD1)

posD2 = 0
while T_data[posD2] < t_max:
    posD2 += 1

print(posD2)

x1= (rho_data[posD1], rho_data[posD2],rho_dataig[posD1], rho_dataig[posD2], rho_nist[posN1],
rho_nist[posN2])
rho_min= round(min(x1)-0.1*abs(max(rho_data)-min(rho_data)), 0)
rho_max= round(max(x1)+0.1*abs(max(rho_data)-min(rho_data)), 0)
x2= (sos_data[posD1], sos_data[posD2],sos_dataig[posD1], sos_dataig[posD2], sos_nist[posN1], sos_nist[posN2])
sos_min= round(min(x2)-0.1*(max(sos_data)-min(sos_data)), 0)
sos_max= round(max(x2)+0.4*(max(sos_data)-min(sos_data)), 0)
x3= (mu_data[posD1], mu_data[posD2],mu_dataig[posD1], mu_dataig[posD2], mu_nist[posN1], mu_nist[posN2])
mu_min= round(min(x3)-0.1*(max(mu_data)-min(mu_data)), 5)
mu_max= round(max(x3)+0.1*(max(mu_data)-min(mu_data)), 5)
x4= (kappa_data[posD1], kappa_data[posD2],kappa_dataig[posD1], kappa_dataig[posD2], kappa_nist[posN1],
kappa_nist[posN2])
kappa_min= round(min(x4)-0.1*(max(kappa_data)-min(kappa_data)), 2)
kappa_max= round(max(x4)+0.1*(max(kappa_data)-min(kappa_data)), 2)

### Plot density vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, rho_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'$\text{NIST}$' )
plt.plot( T_data, rho_data, linestyle = '--', color = 'firebrick', label = r'$\text{RHEA}$' )
plt.plot( T_dataig, rho_dataig, linestyle = ':', color = 'firebrick', label = r'$\text{IDEAL GAS}$' )
# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = 'True', direction = 'inout')
plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 25 ) )
#plt.xscale( 'log' )
plt.xlabel( r'$T$ \thinspace \text{[K]}$' )
plt.ylim( rho_min, rho_max )
plt.yticks( np.arange( rho_min, rho_max + 50, 150 ) )
#plt.yscale( 'log' )
plt.ylabel( r'$\rho$ \thinspace \text{[kg/m]}^3 \text{[g/cm]}^3$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )

```

```

plt.savefig( 'density_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'density_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

### Plot internal energy vs. temperature

offsetE=e_nist[posN1]-e_data[posD1]/1.0e3;

E_min= round(E_data[0]/1.0e3+offsetE-abs(0.1*(abs(E_data[-1]/1.0e3)-abs(E_data[0]/1.0e3))), 0)
E_max= round(E_data[-1]/1.0e3+offsetE+abs(0.1*(abs(E_data[-1]/1.0e3)-abs(E_data[0]/1.0e3))), 0)

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, e_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\texttrm{NIST}' )
plt.plot( T_data, e_data/1.0e3 + offsetE, linestyle = '--', color = 'firebrick', label = r'\texttrm{RHEA}' )
plt.plot( T_dataig, e_dataig/1.0e3, linestyle = ':', color = 'firebrick', label = r'\texttrm{IDEAL GAS}' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 25 ) )
plt.xscale( 'log' )
plt.xlabel( r'$T \text{ \thinspace } \texttrm{[K]}' )
plt.ylim( E_min, E_max )
plt.yticks( np.arange( E_min, E_max + 10, 25 ) )
plt.yscale( 'log' )
plt.ylabel( r'$e \text{ \thinspace } \texttrm{[kJ/kg]}' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='lower right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'internal_energy_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'internal_energy_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

### Plot speed of sound vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, sos_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\texttrm{NIST}' )
plt.plot( T_data, sos_data, linestyle = '--', color = 'firebrick', label = r'\texttrm{RHEA}' )
plt.plot( T_dataig, sos_dataig, linestyle = ':', color = 'firebrick', label = r'\texttrm{IDEAL GAS}' )
# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 25 ) )
plt.xscale( 'log' )
plt.xlabel( r'$T \text{ \thinspace } \texttrm{[K]}' )
plt.ylim( sos_min, sos_max )
plt.yticks( np.arange( sos_min, sos_max + 10, 100 ) )
plt.yscale( 'log' )
plt.ylabel( r'$c \text{ \thinspace } \texttrm{[m/s]}' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'sound_speed_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'sound_speed_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

### Plot viscosity vs. temperature

# Clear plot
plt.clf()

```

```

# Read & Plot data
plt.scatter( T_nist, mu_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\textrm{NIST}$' )
plt.plot( T_data, mu_data, linestyle = '--', color = 'firebrick', label = r'\textrm{RHEA}$' )
plt.plot( T_dataig, mu_dataig, linestyle = ':', color = 'firebrick', label = r'\textrm{IDEAL GAS}$' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.ticklabel_format(axis = 'y', style = 'sci', scilimits=(0,0))
plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 25 ) )
plt.xscale( 'log' )
plt.xlabel( r'$T \thinspace \textrm{[K]}$' )
plt.ylim( mu_min, mu_max )
plt.yticks( np.arange( mu_min, mu_max + 0.00001, 0.1 ) )
plt.yscale( 'log' )
plt.ylabel( r'$\mu \thinspace \textrm{[Pa]}\cdot\textrm{[s]}$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'viscosity_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'viscosity_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

### Plot thermal conductivity vs. temperature

# Clear plot
plt.clf()

# Read & Plot data
plt.scatter( T_nist, kappa_nist, s = 25, facecolors = 'none', edgecolors = 'black', label = r'\textrm{NIST}$' )
plt.plot( T_data, kappa_data, linestyle = '--', color = 'firebrick', label = r'\textrm{RHEA}$' )
plt.plot( T_dataig, kappa_dataig, linestyle = ':', color = 'firebrick', label = r'\textrm{IDEAL GAS}$' )

# Configure plot
plt.tick_params(axis = 'both', top = True, bottom = True, right = 'True', left = True, direction = 'inout')
plt.ticklabel_format(axis = 'y', style = 'sci', scilimits=(0,0))
plt.xlim( t_min, t_max )
plt.xticks( np.arange( t_min, t_max + 10, 25 ) )
plt.xscale( 'log' )
plt.xlabel( r'$T \thinspace \textrm{[K]}$' )
plt.ylim( kappa_min, kappa_max )
plt.yticks( np.arange( kappa_min, kappa_max + 0.01, 0.5 ) )
plt.yscale( 'log' )
plt.ylabel( r'$\kappa \thinspace \textrm{[W/(m)]}\cdot\textrm{[K]}$' )
legend = plt.legend( shadow = False, fancybox = False, frameon = False, loc='upper right' )
plt.tick_params( axis = 'both', pad = 7.5 )
plt.savefig( 'thermal_conductivity_vs_temperature.eps', format = 'eps', bbox_inches = 'tight' )
plt.savefig( 'thermal_conductivity_vs_temperature.png', format = 'png', bbox_inches = 'tight' )

```

A6. Programa Scilab

```

function RepresentaBB()

    Tc=154.58
    Pc=5043000
    omega=0.0222
    Tini=100
    Tfi=250

    CalculBB(Tc,Pc,omega,Tini,Tfi,82,"Res1.txt","r-")

endfunction

function CalculBB(Tc, Pc, omega, Tini, Tfi, P, nomRes, colGraf)

    R=8314
    p=P*1000000

    a=0.45723553*R^2*Tc^2/Pc
    b=0.07779607*R*Tc/Pc

    kappa=0.37464+1.5422*omega-0.26993*omega^2

    pos=1
    for T=Tini:1:Tfi,
        alfa=(1+kappa*(1-(T/Tc)^0.5))^2
        a1= b-R*T/p
        a2= alfa*a/p-3*b^2-2*R*T*b/p
        a3= b^3+R*T*b^2/p-alfa*a*b/p

        b1= (3*a2-a1^2)/3
        b2= (2*a1^3-9*a1*a2+27*a3)/27

        T_vec(pos)=T
        BB_vec(pos)=b2^2/4+b1^3/27
        CC=nthroot((-b2/2+(BB_vec(pos))^0.5),3)
        DD=nthroot((-b2/2-(BB_vec(pos))^0.5),3)
        C_vec(pos)=CC
        D_vec(pos)=DD

        vol_vec= (CC+DD-a1/3)/31.999
        den_vec(pos)=1/vol_vec;
        pos=pos+1
    end

    //Per representar el valor del coeficient
    scf(1)
    plot(T_vec,BB_vec,colGraf)
    xgrid(3)
    matRes=[T_vec,BB_vec]
    fprintfMat(nomRes+"_coef",matRes)

    //Per representar el valor de volum específic
    scf(2)
    plot(T_vec,den_vec,colGraf)
    xgrid(3)
    matRes2=[T_vec,den_vec]
    fprintfMat(nomRes+"_den",matRes2)

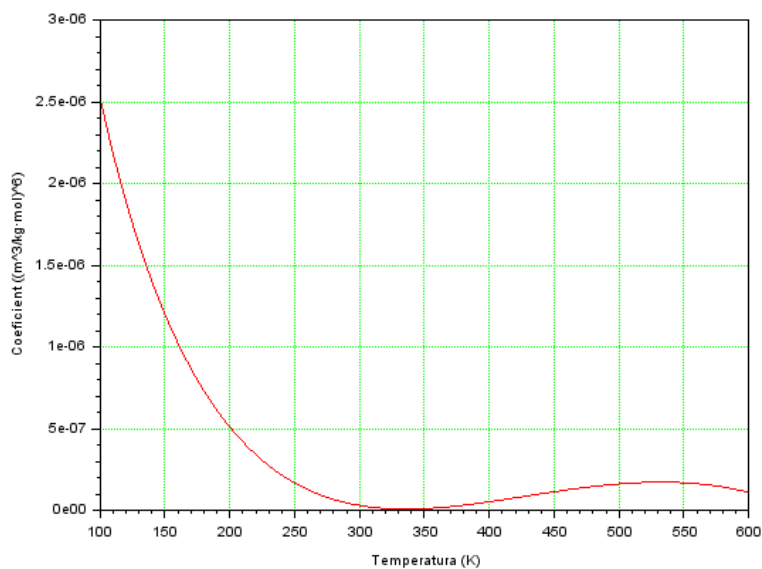
endfunction

```

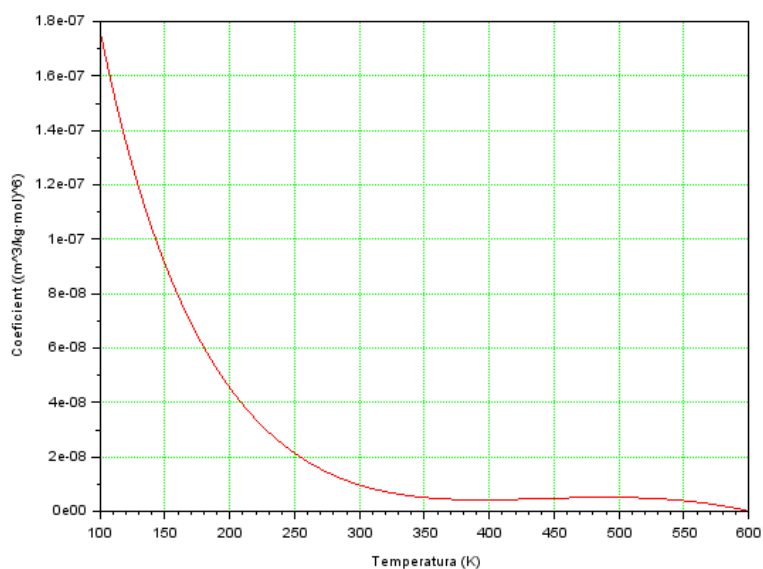

Annex B: Gràfiques Scilab

En aquest Annex es mostren les gràfiques del programa Scilab, que mostren el resultat del coeficient en funció de la temperatura.

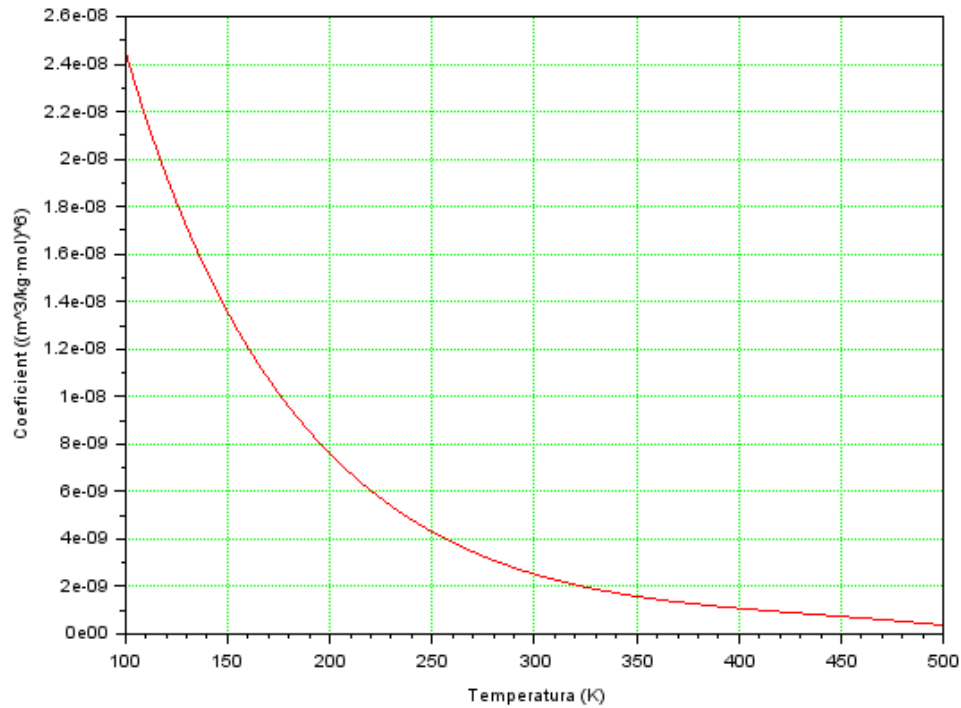
Diòxid de carboni (14,75 MPa)



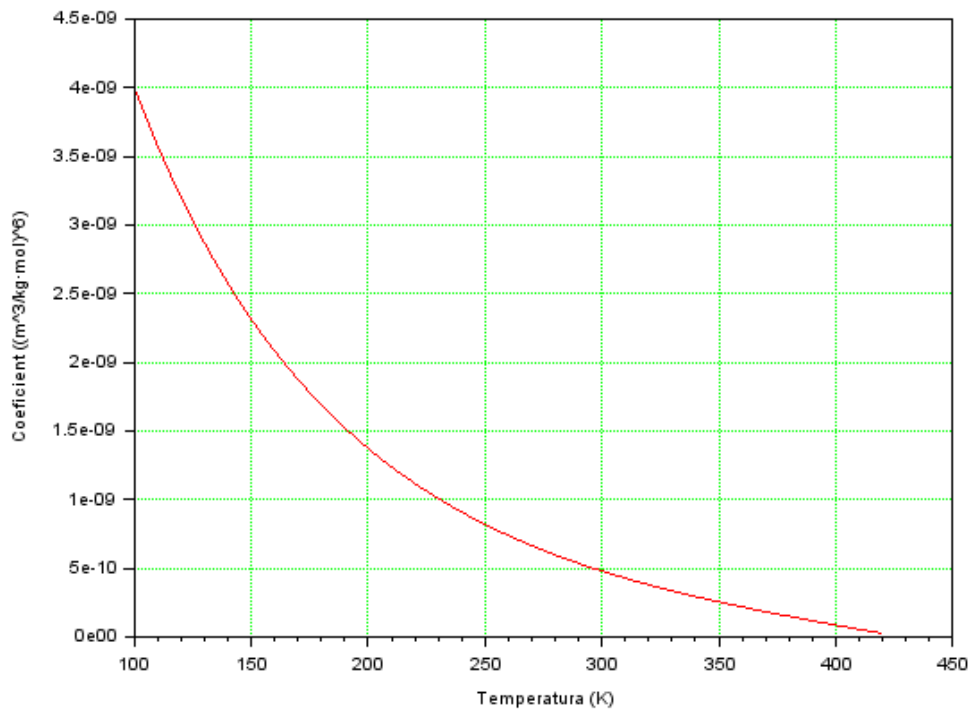
Diòxid de carboni (36,89 MPa)



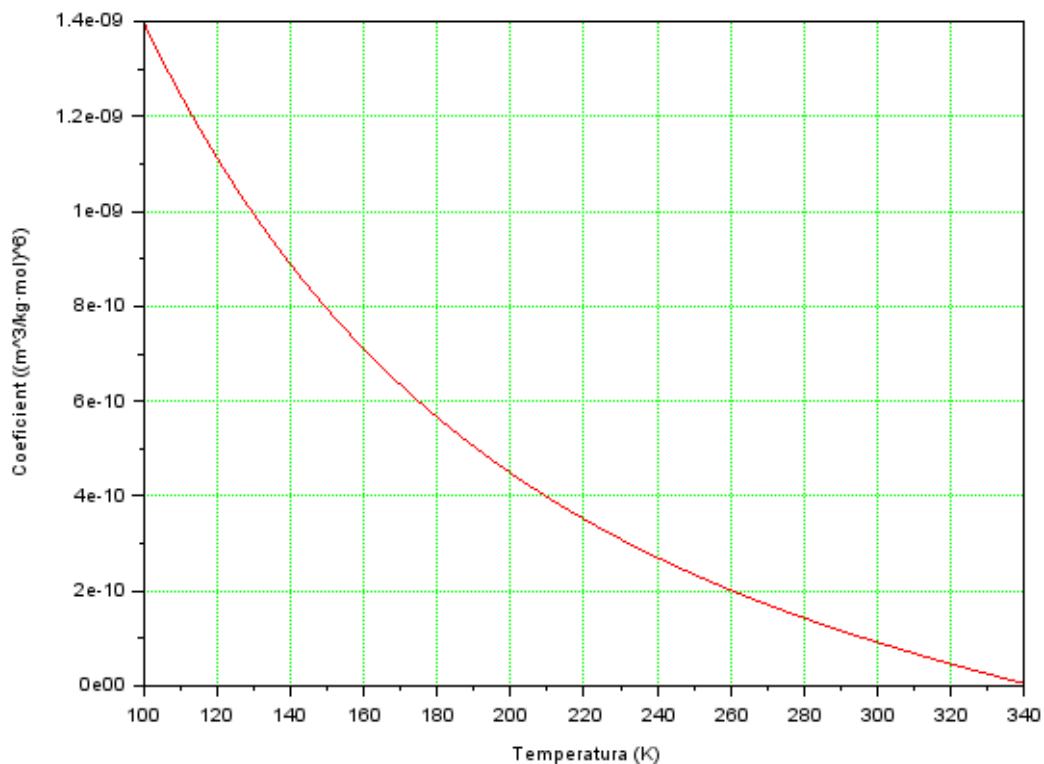
Diòxid de carboni (73,77 MPa)



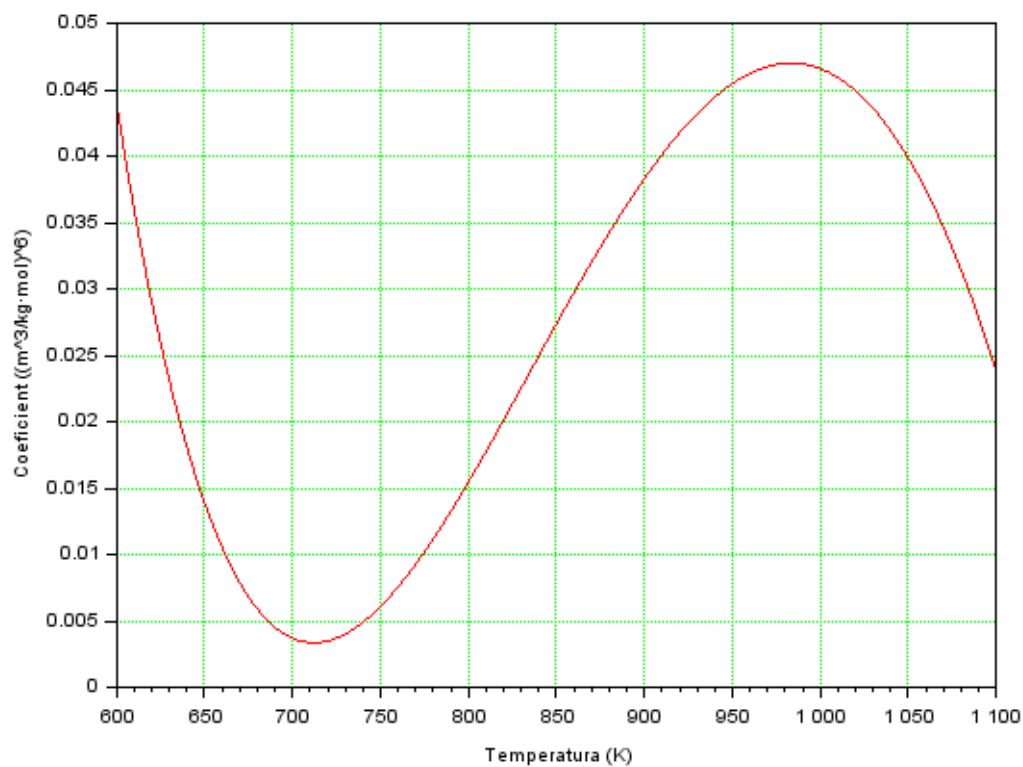
Diòxid de carboni (140 MPa)



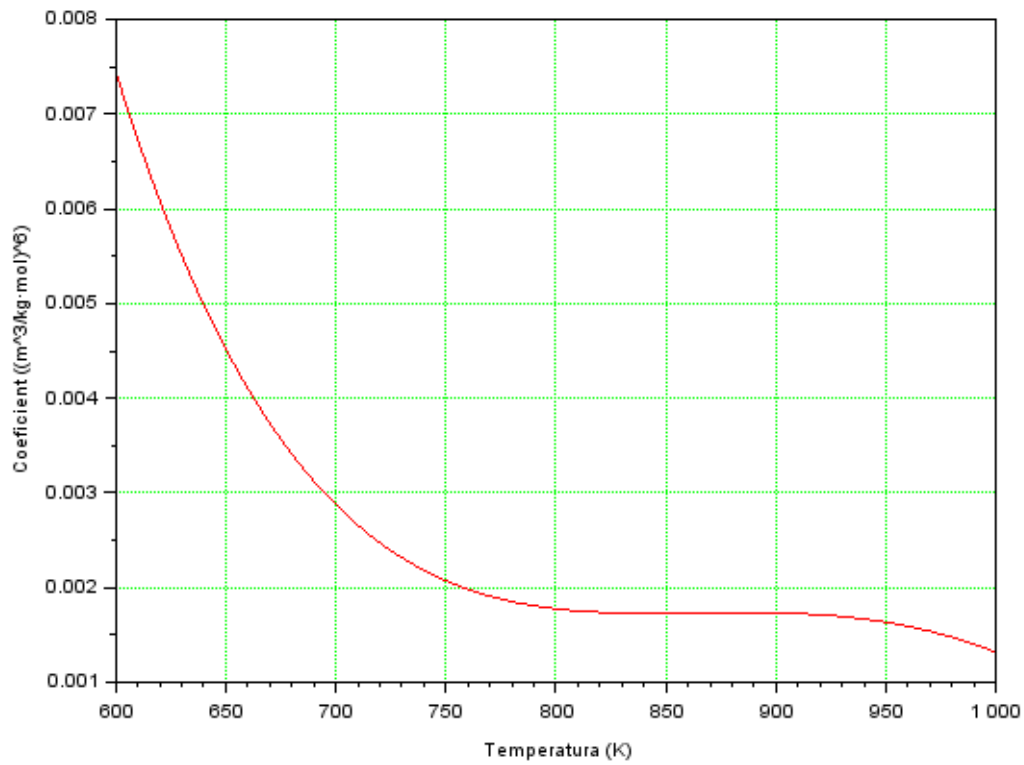
Diòxid de carboni (200 MPa)



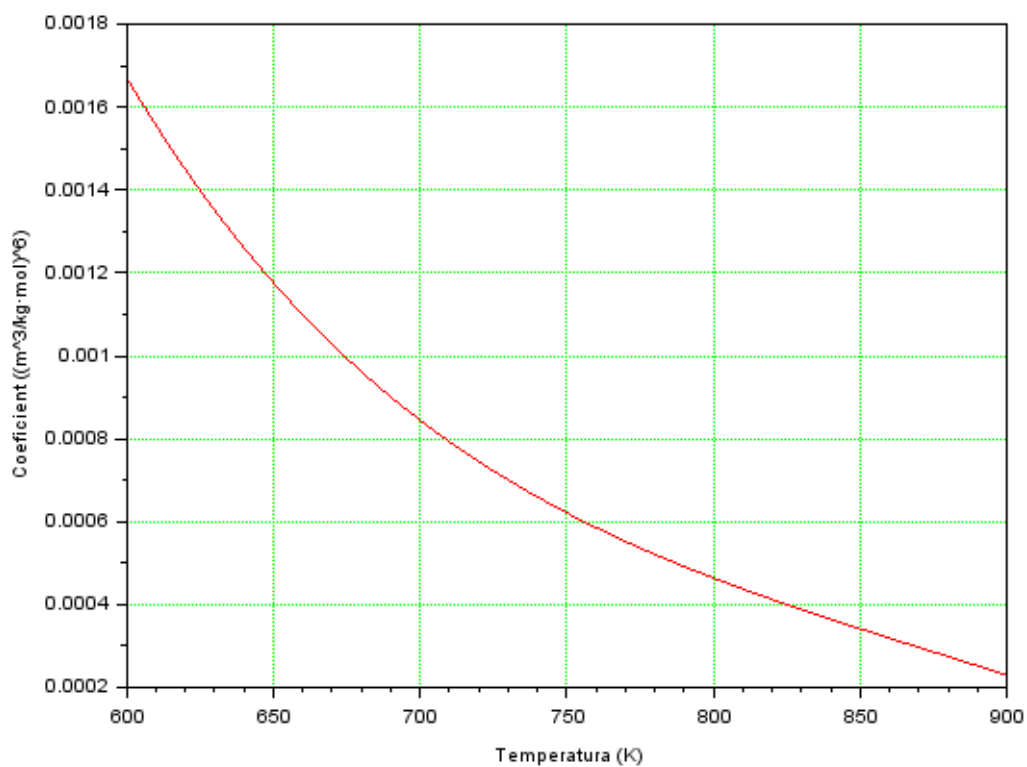
Dodecà (3,63 MPa)



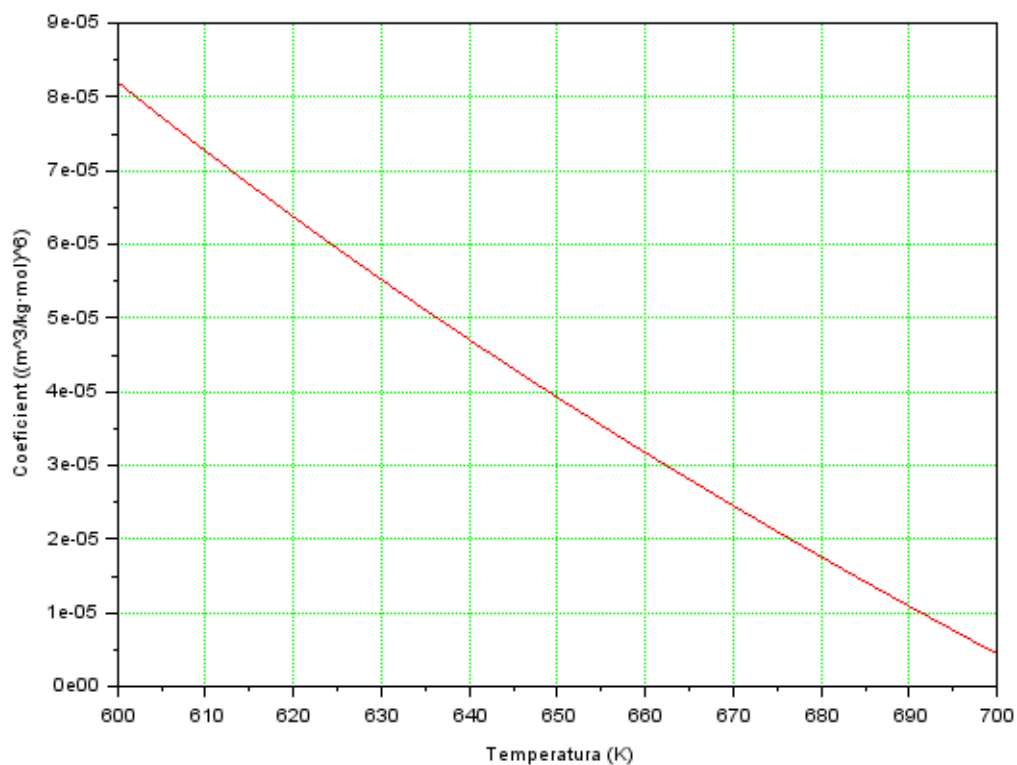
Dodecà (9,09 MPa)



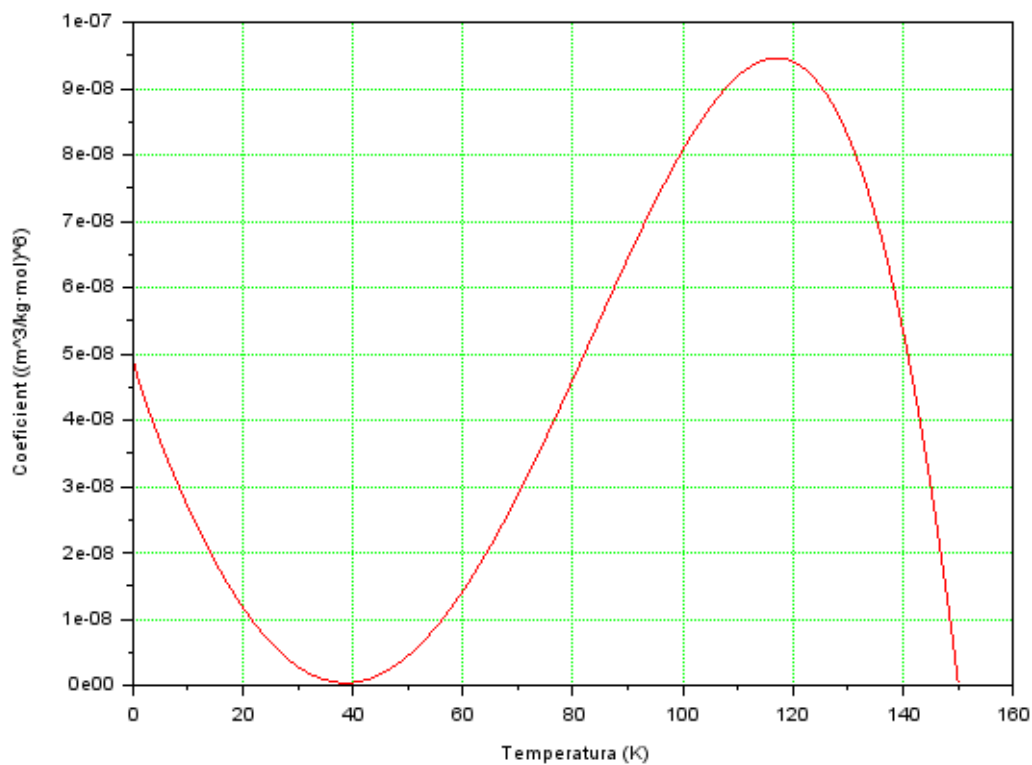
Dodecà (18,17 MPa)



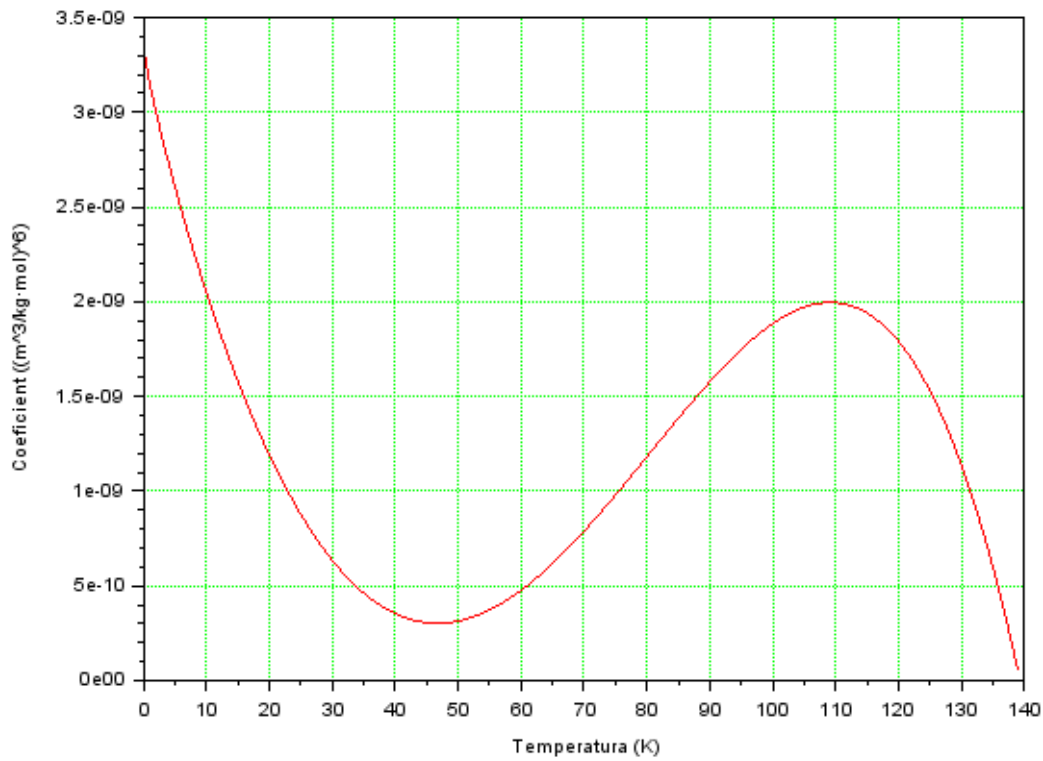
Dodecà (50 MPa)



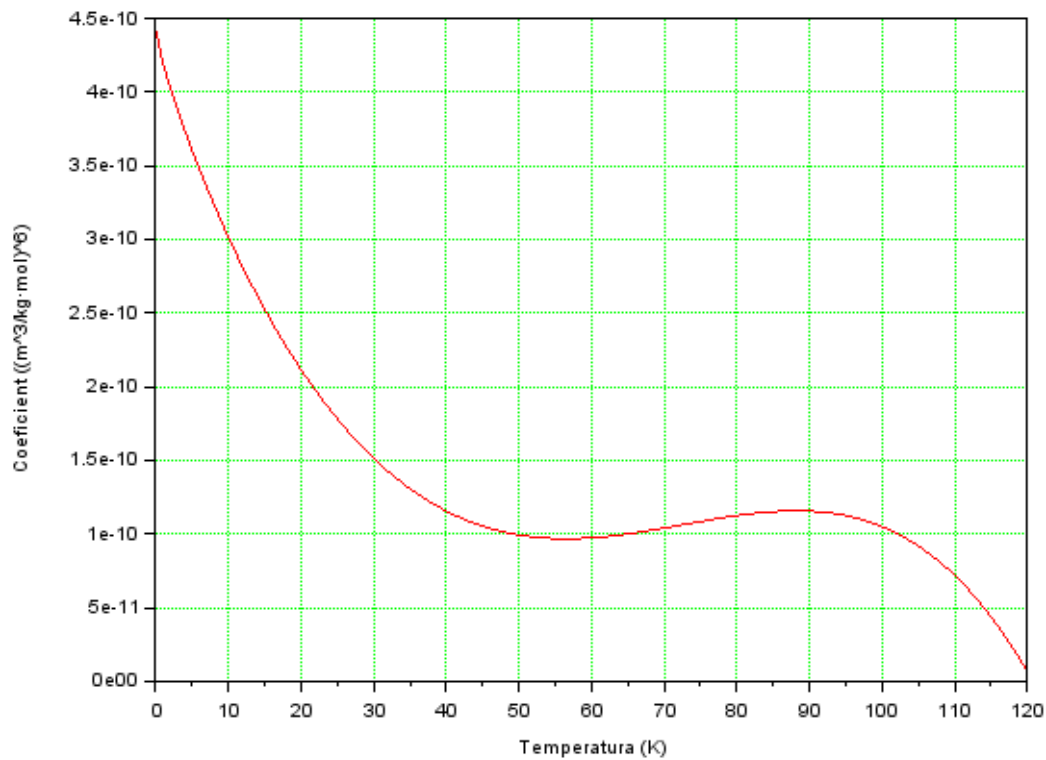
Hidrogen (2,59 MPa)



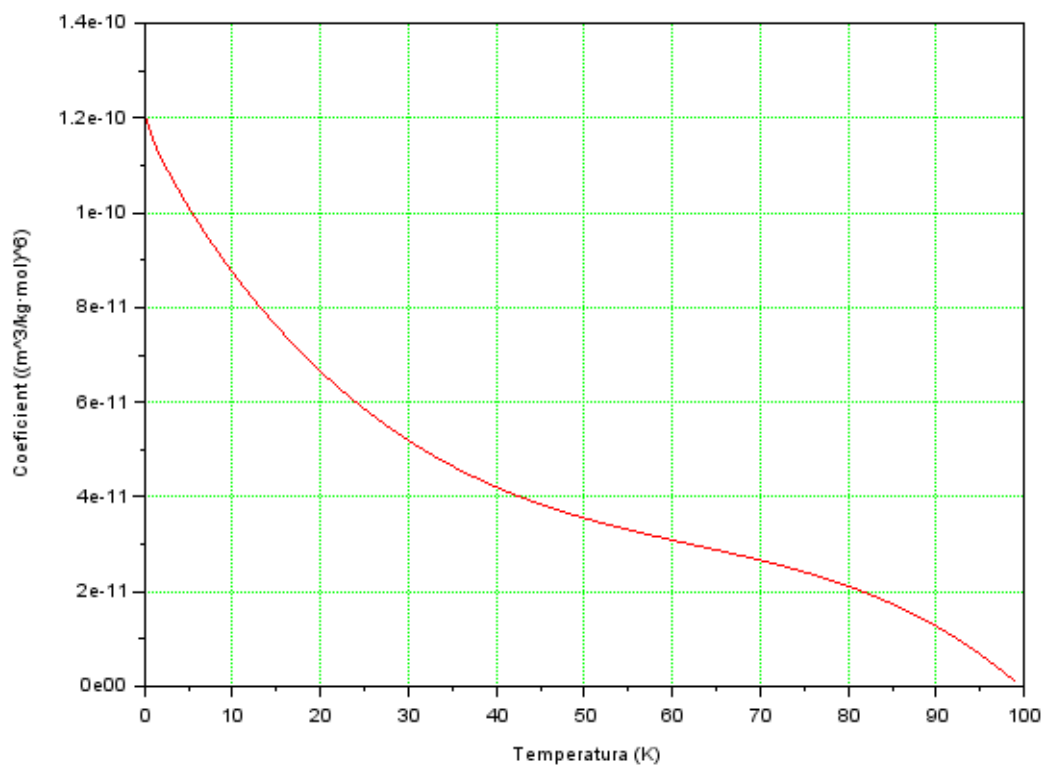
Hidrogen (6,48 MPa)



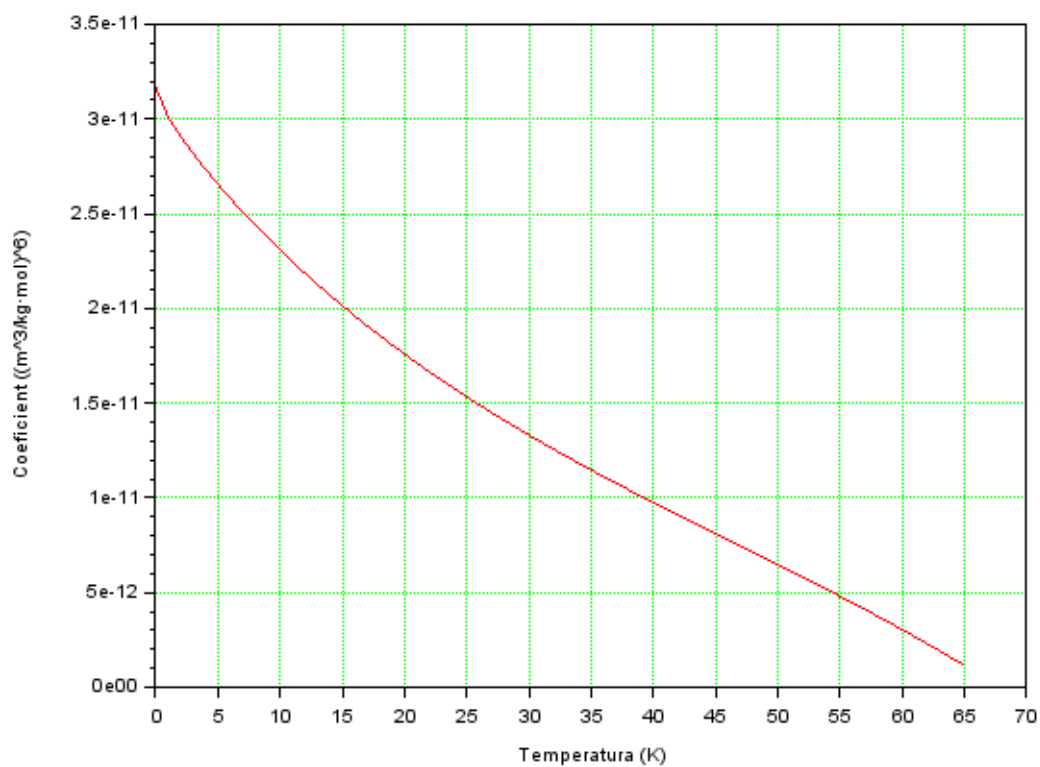
Hidrogen (12,96 MPa)



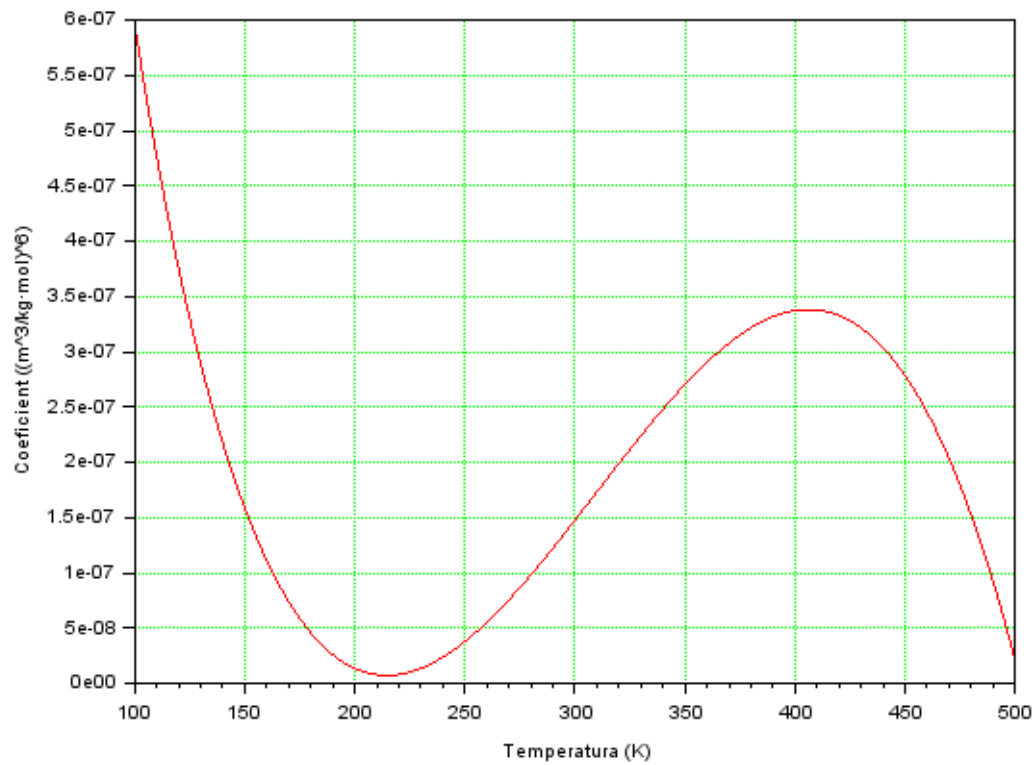
Hidrogen (20 MPa)



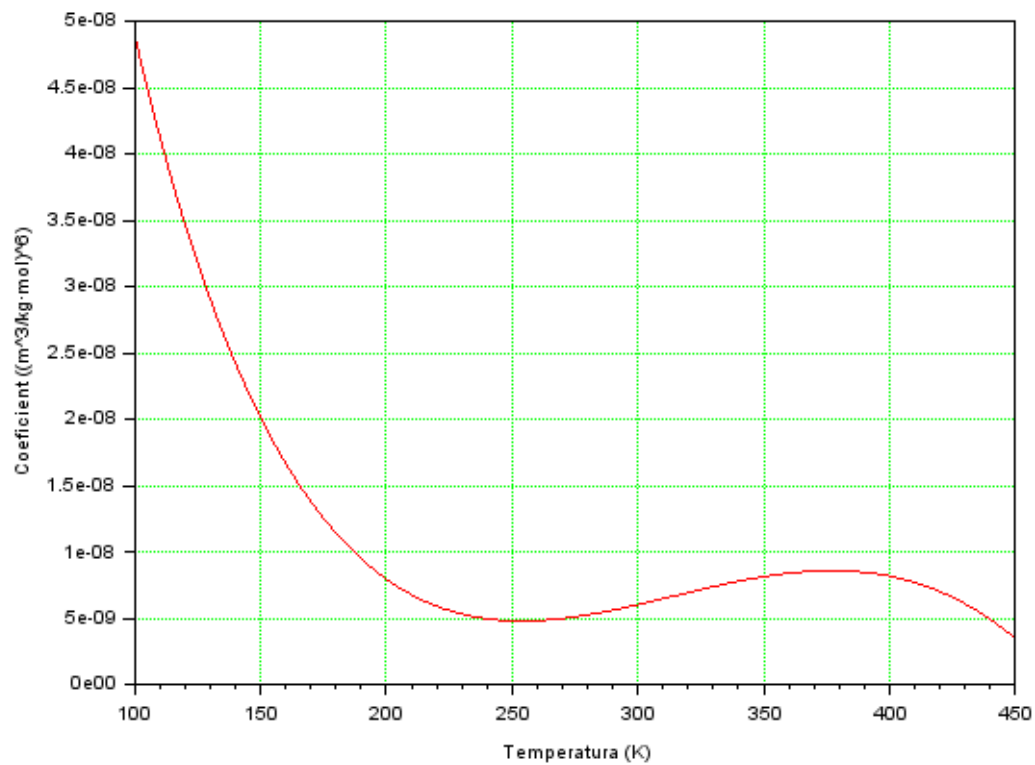
Hidrogen (30 MPa)



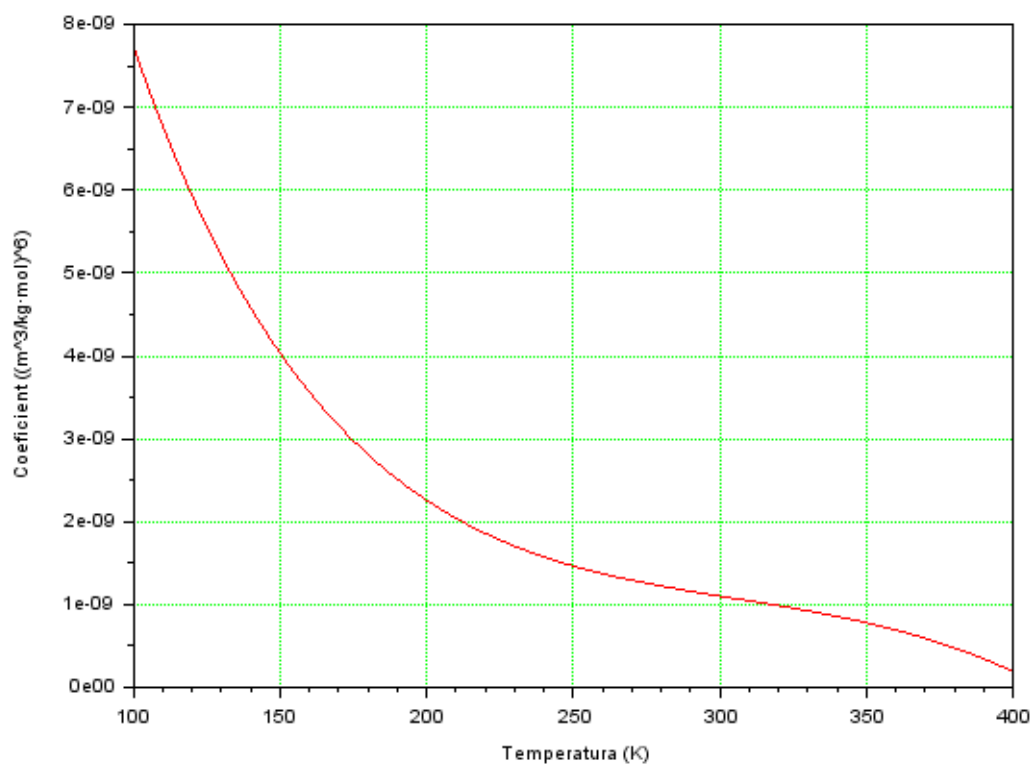
Metà (9,19 MPa)



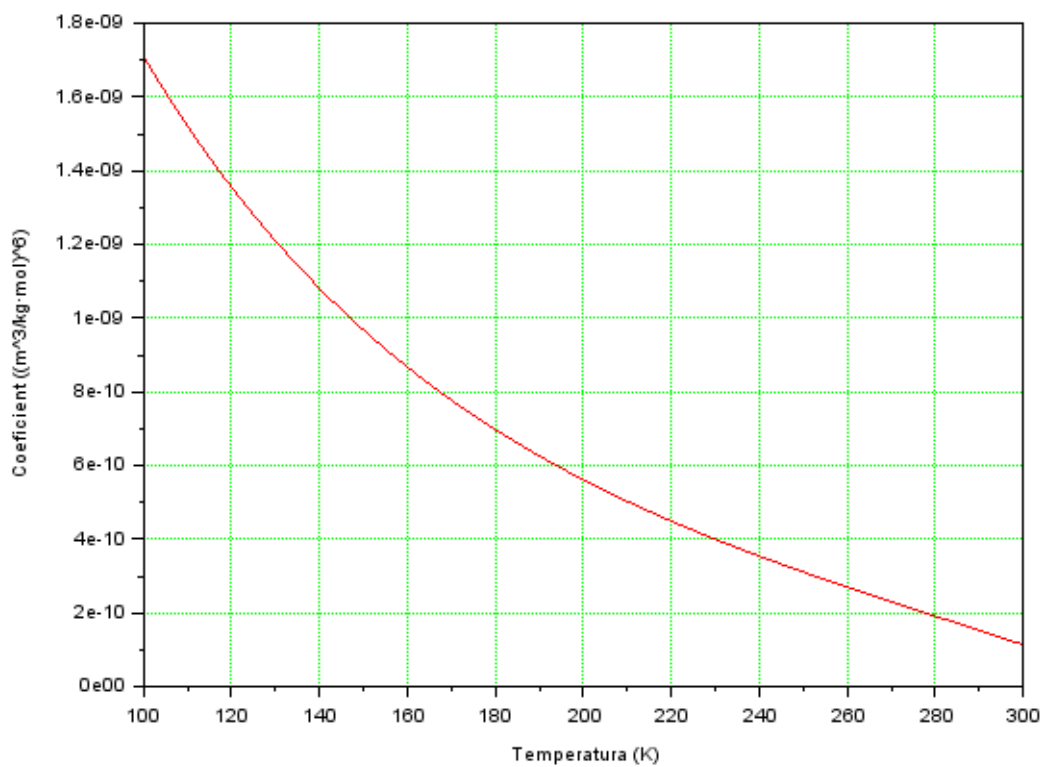
Metà (22,99 MPa)



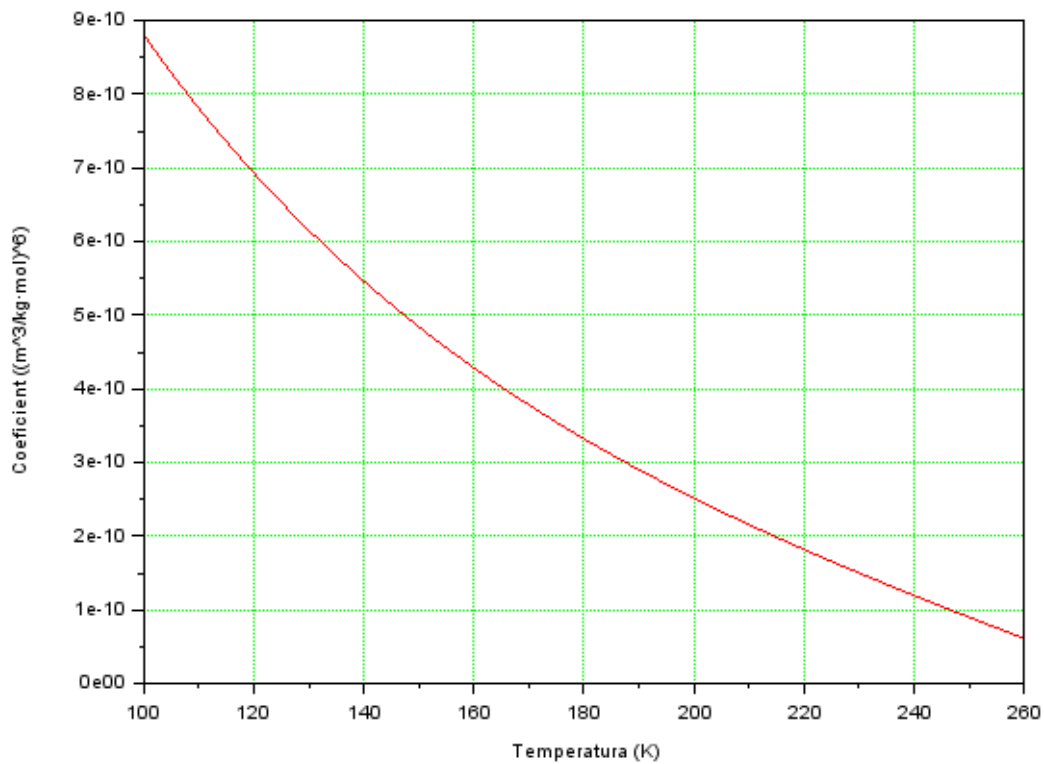
Metà (45,99 MPa)



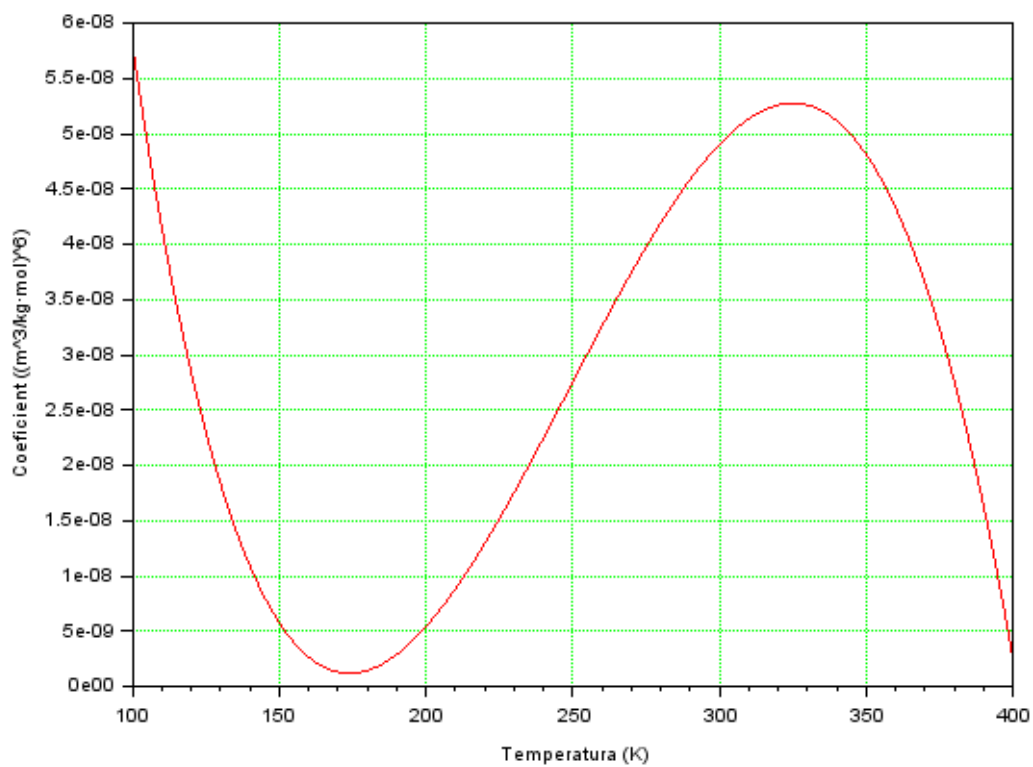
Metà (80 MPa)



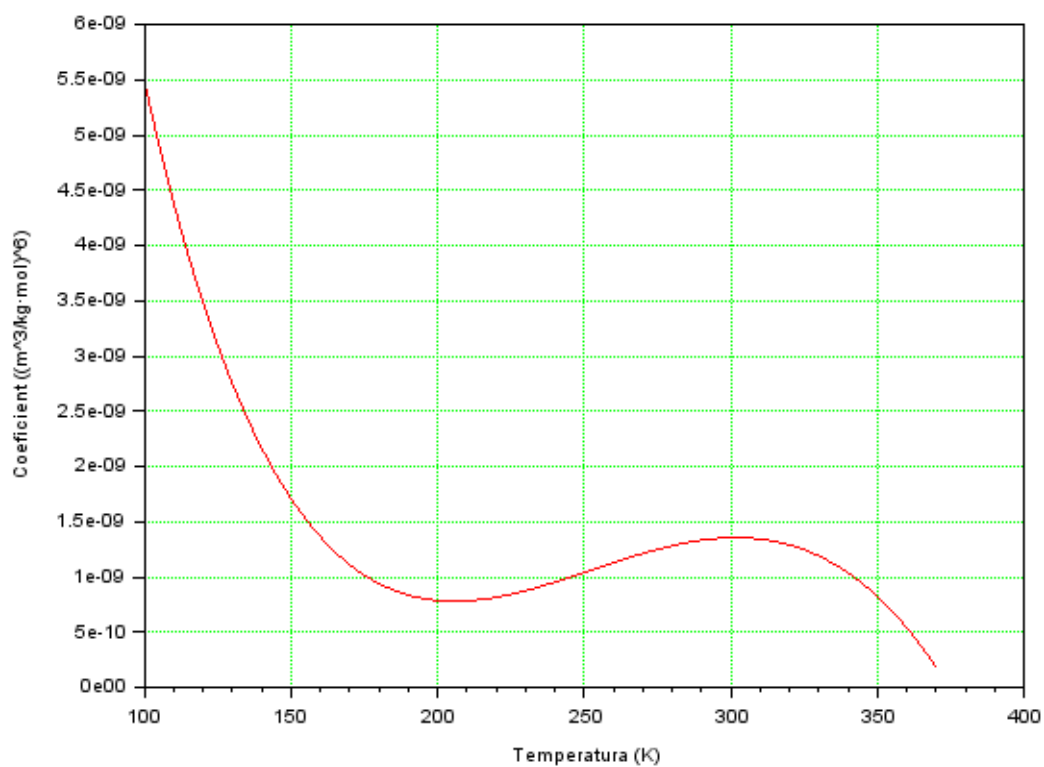
Metà (100 MPa)



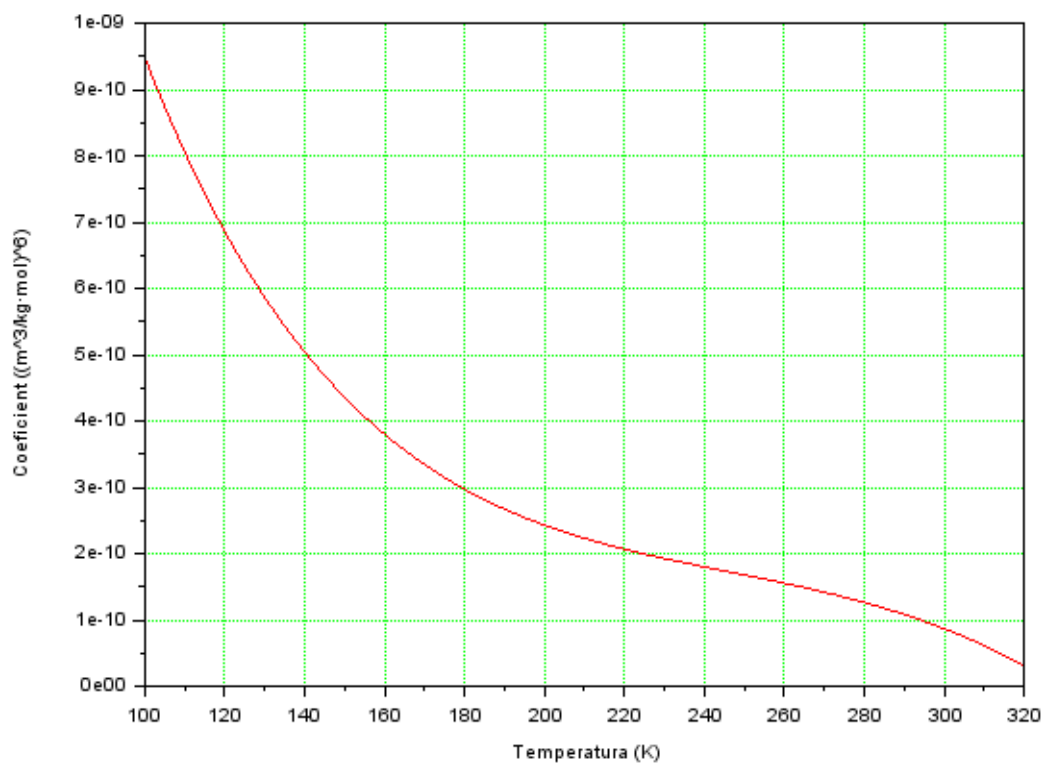
Oxigen (10,09 MPa)



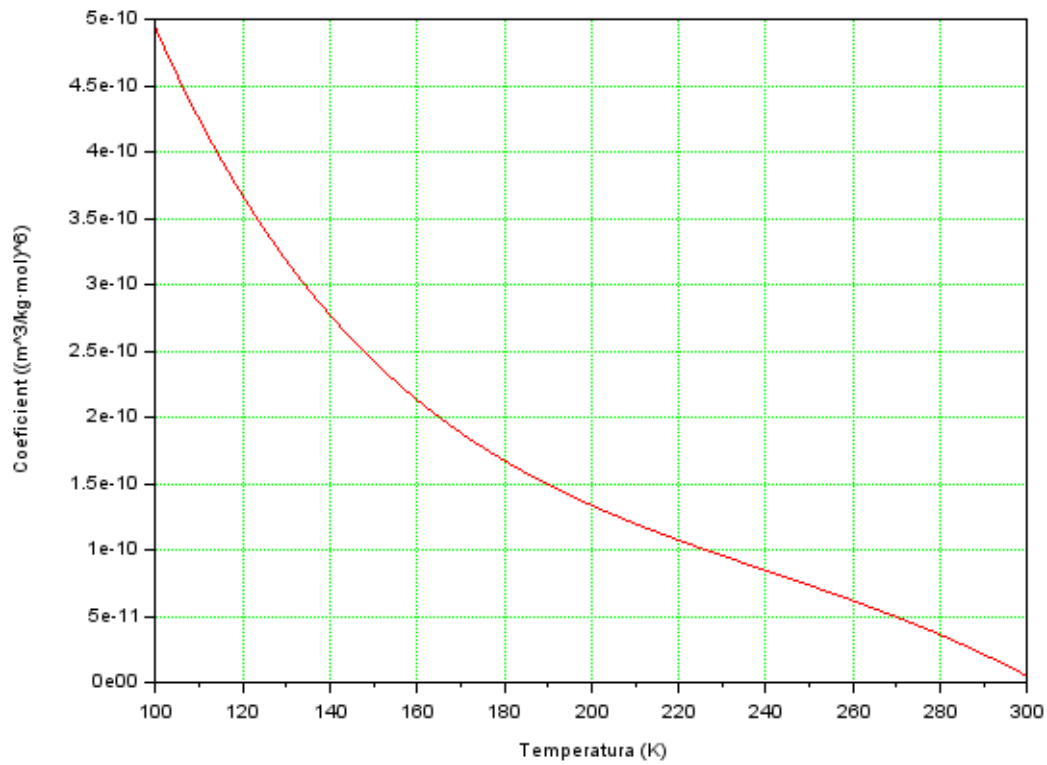
Oxigen (25,22 MPa)



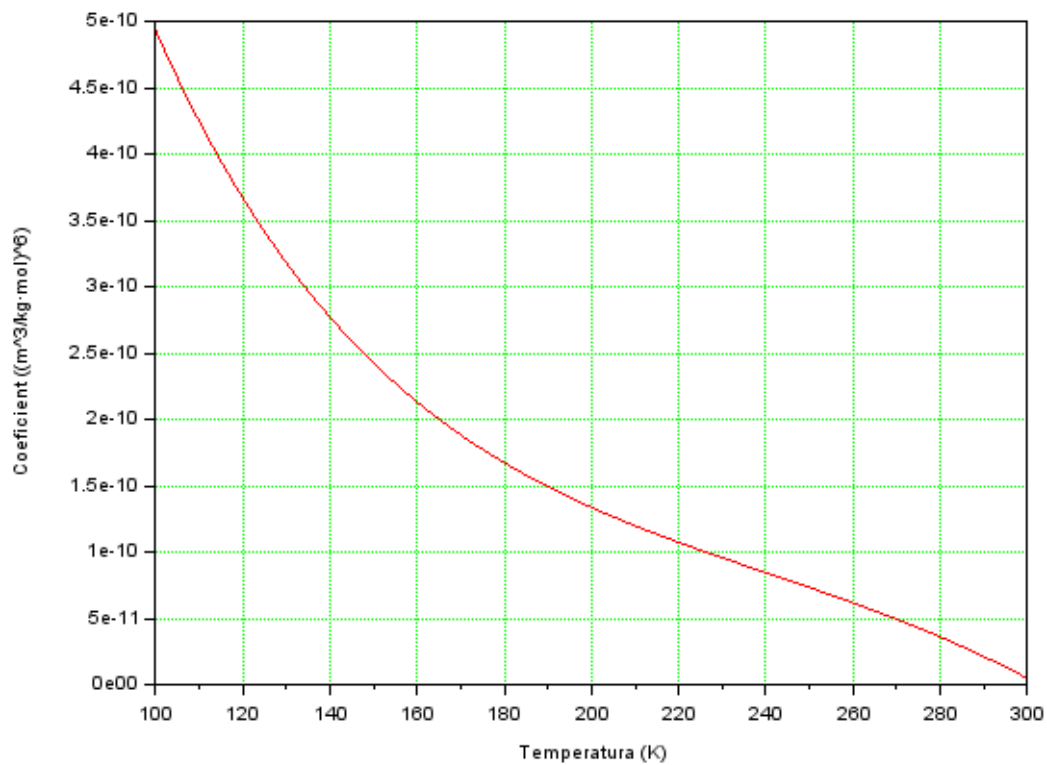
Oxigen (50,43 MPa)



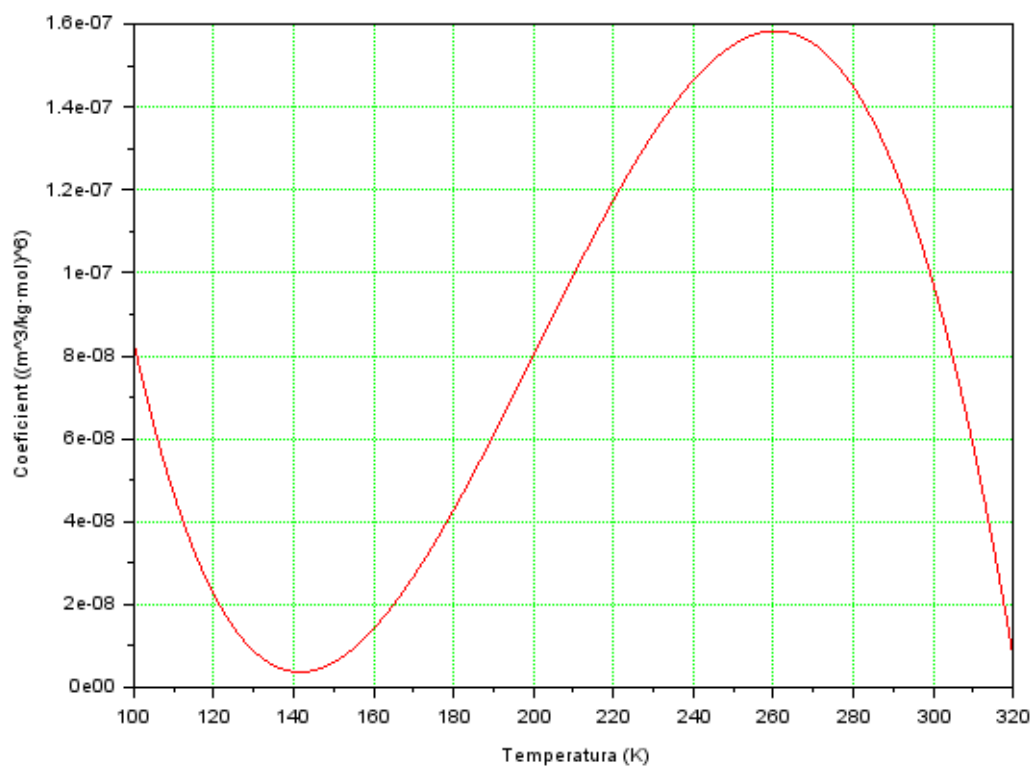
Oxigen (65 MPa)



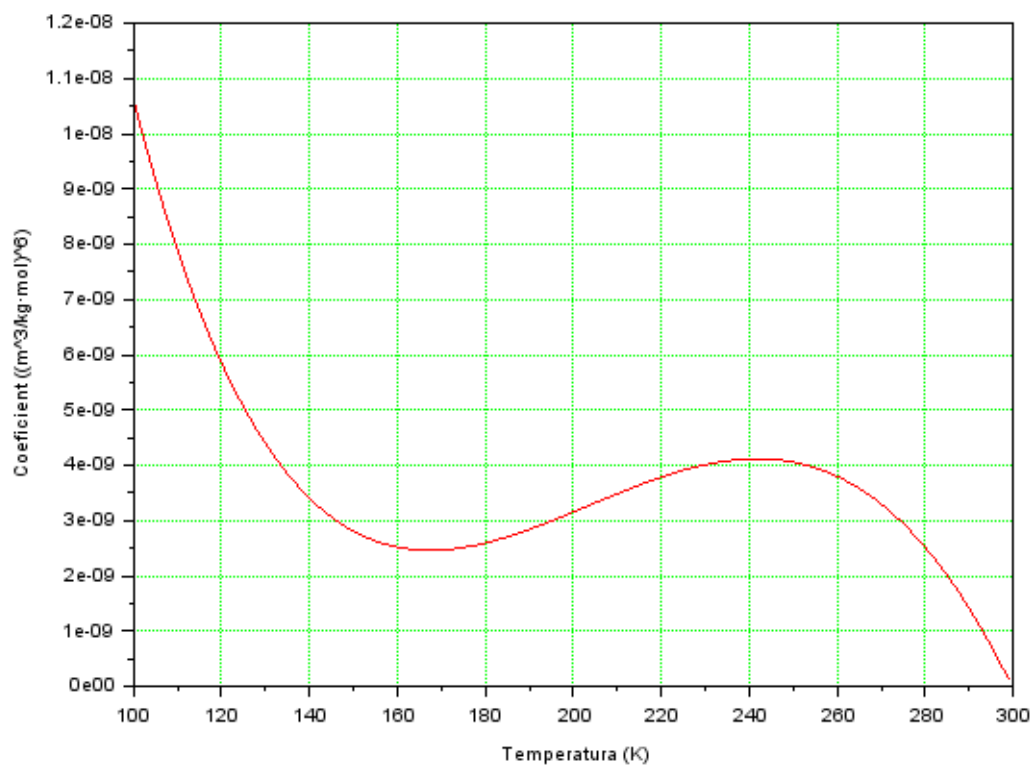
Oxigen (82 MPa)



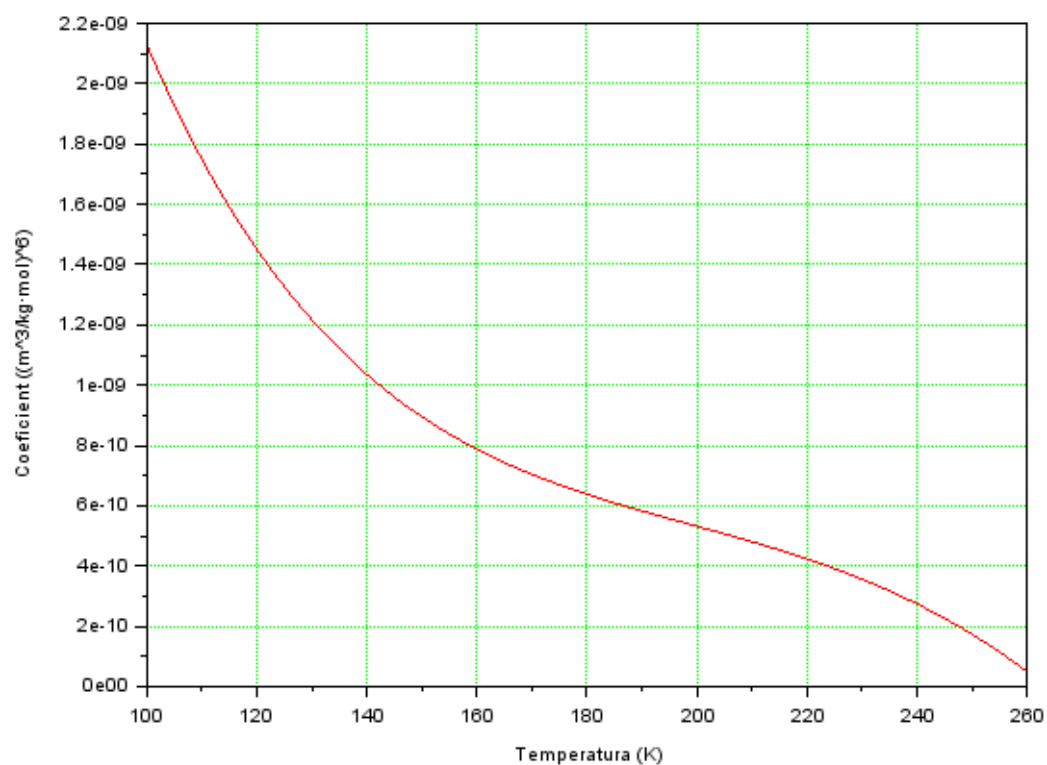
Nitrogen (6,79 MPa)



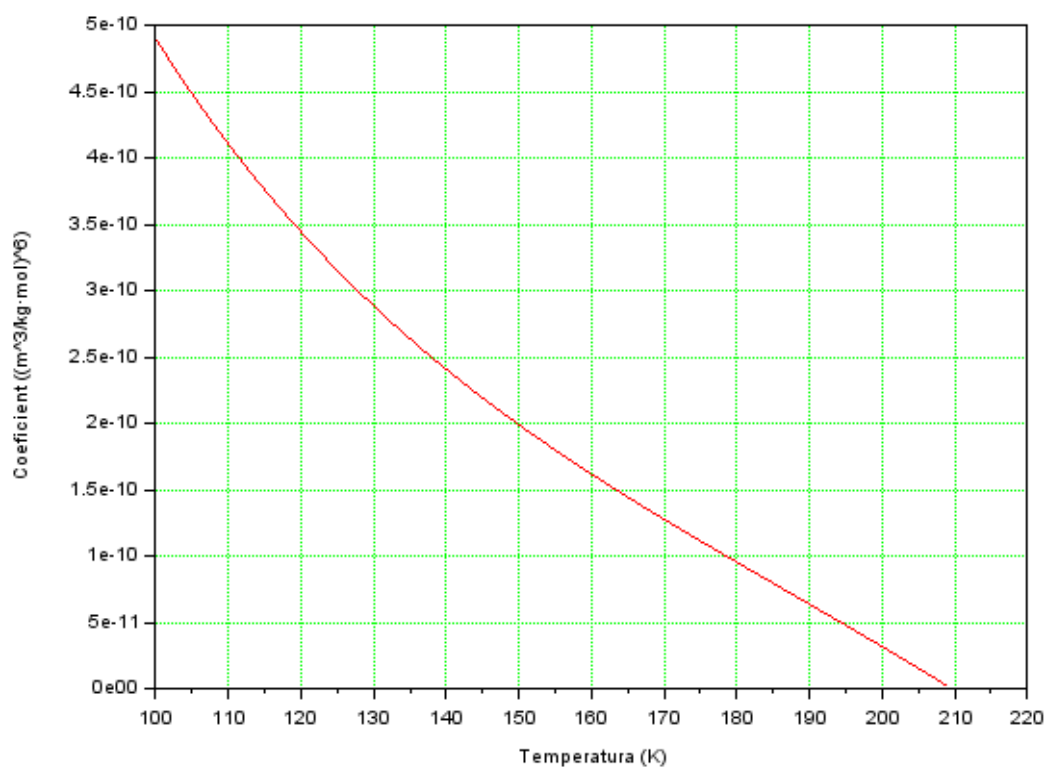
Nitrogen (16,97 MPa)



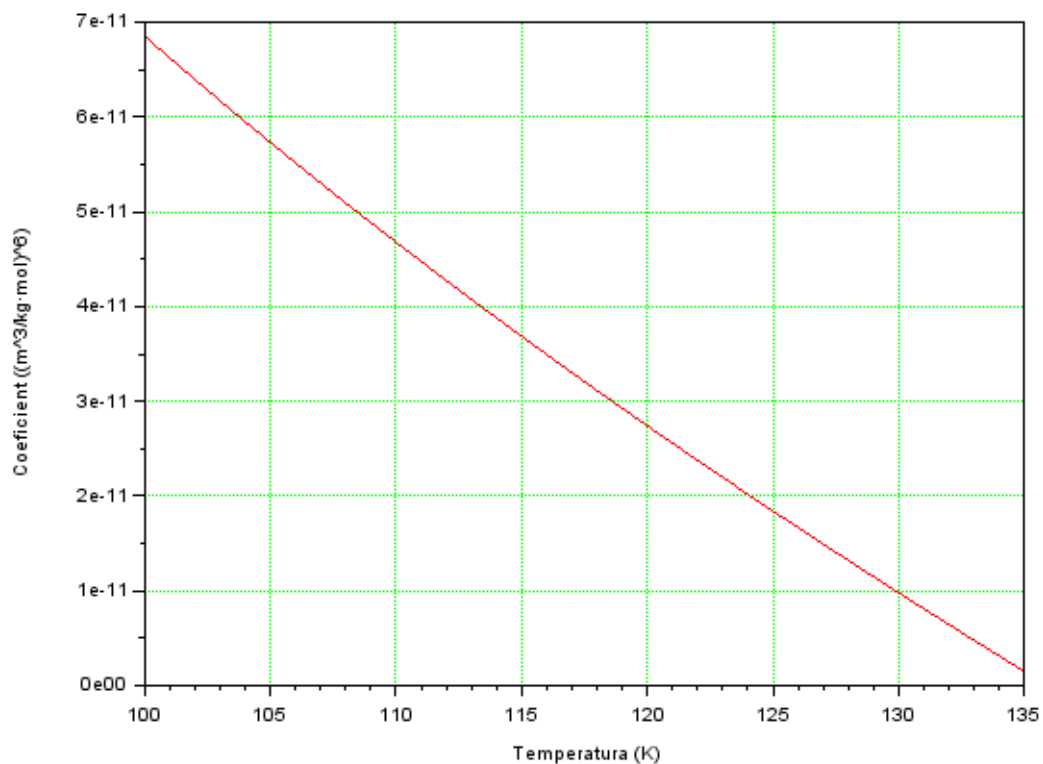
Nitrogen (33,95 MPa)



Nitrogen (60 MPa)



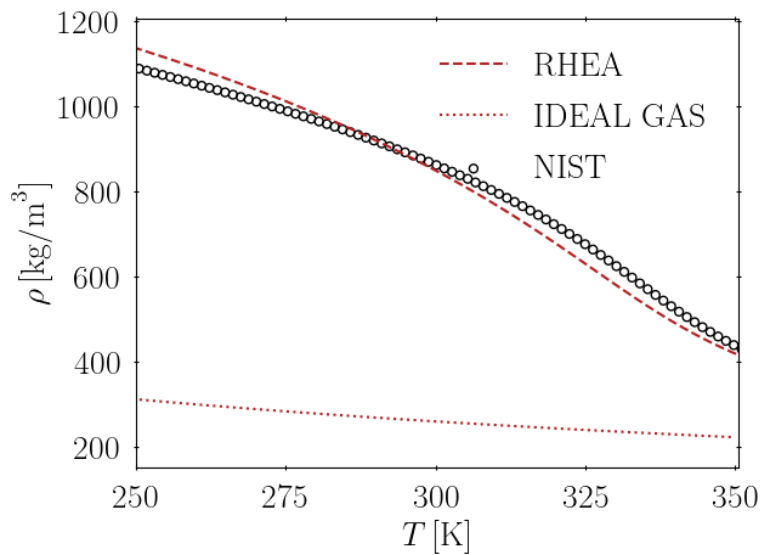
Nitrogen (100 MPa)



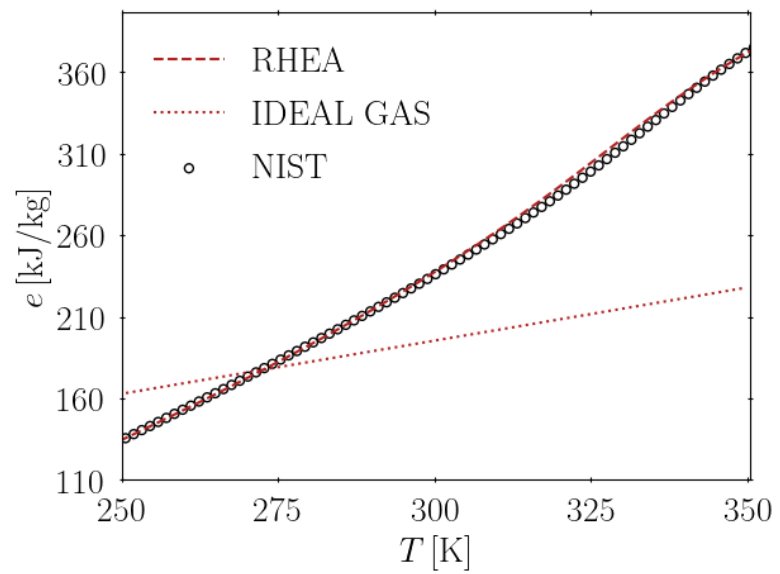
Annex C: Gràfiques RHEA

Diòxid de carboni (14,75 MPa)

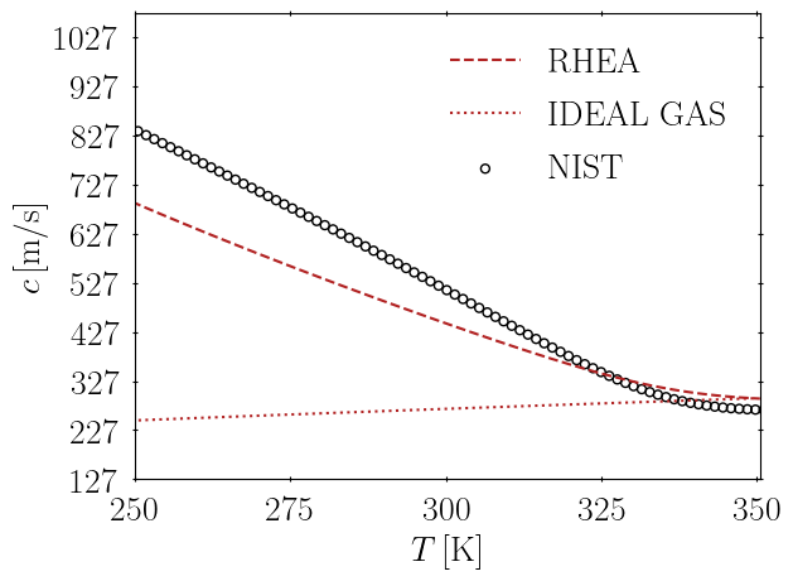
Density vs Temperature



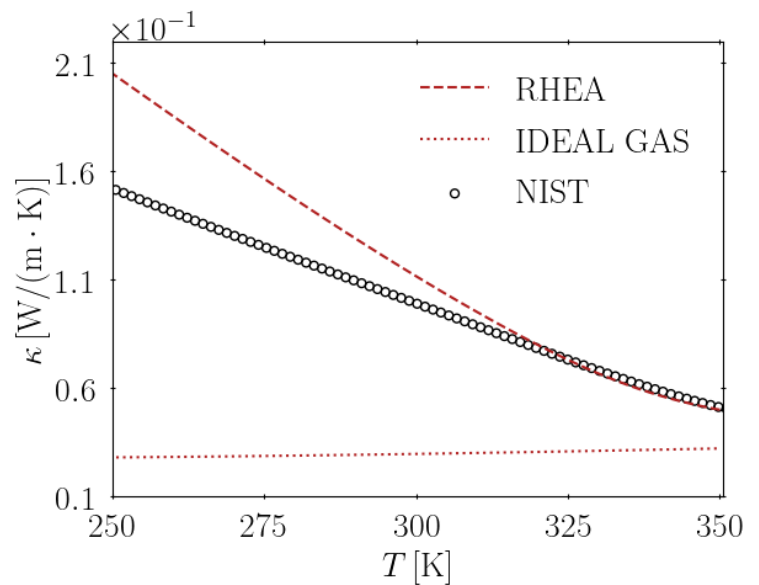
Internal Energy vs Temperature



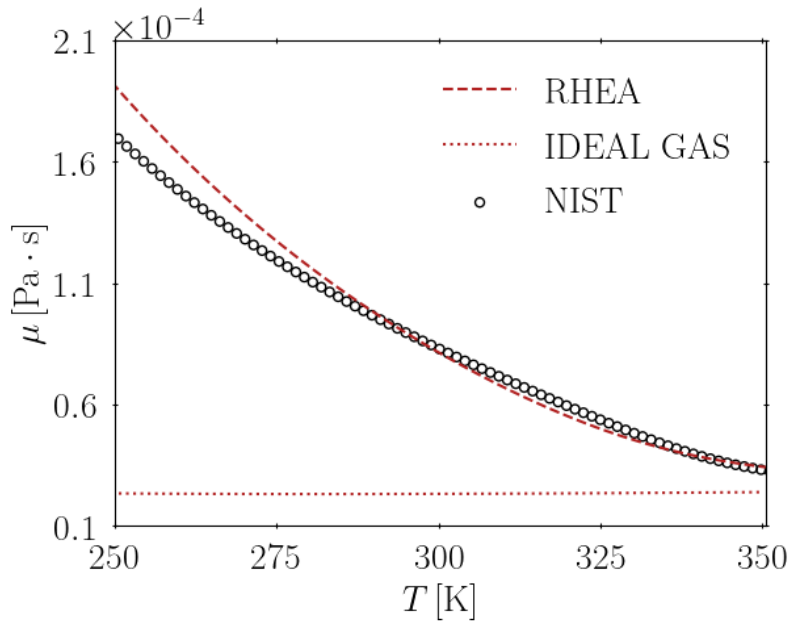
Sound speed vs Temperature



Thermal conductivity vs temperature

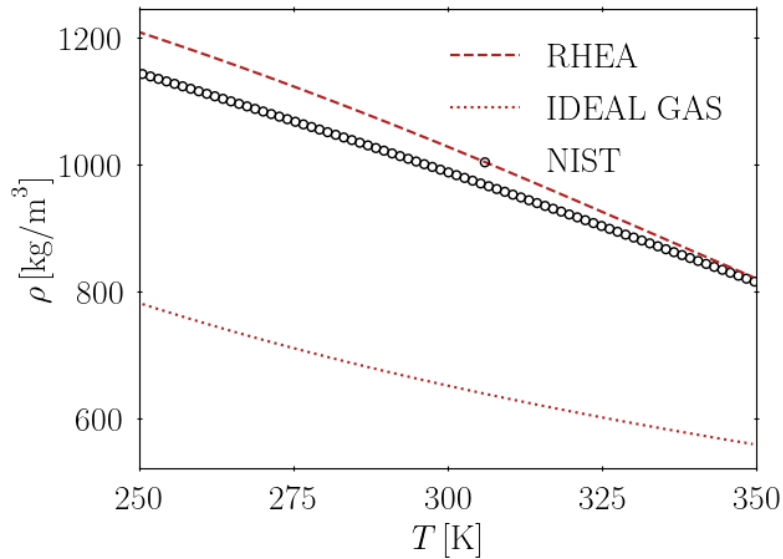


Viscosity vs Temperature

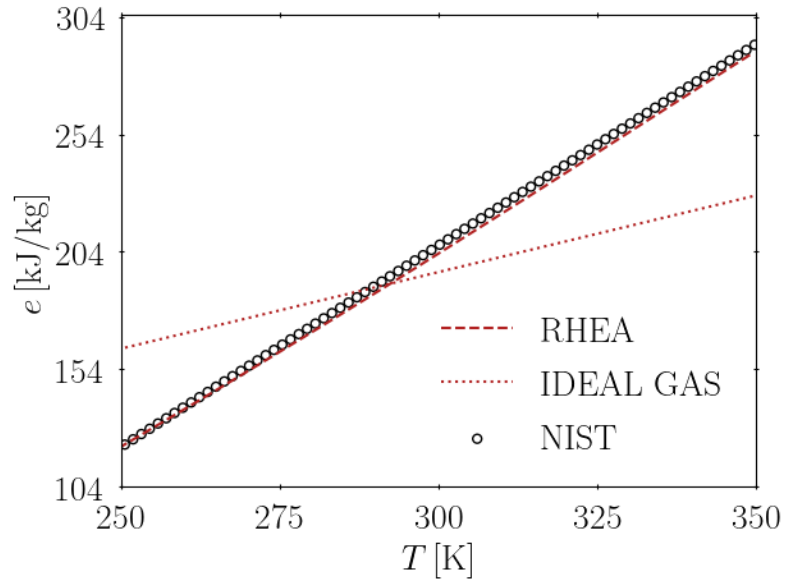


Diòxid de carboni (36,89 MPa)

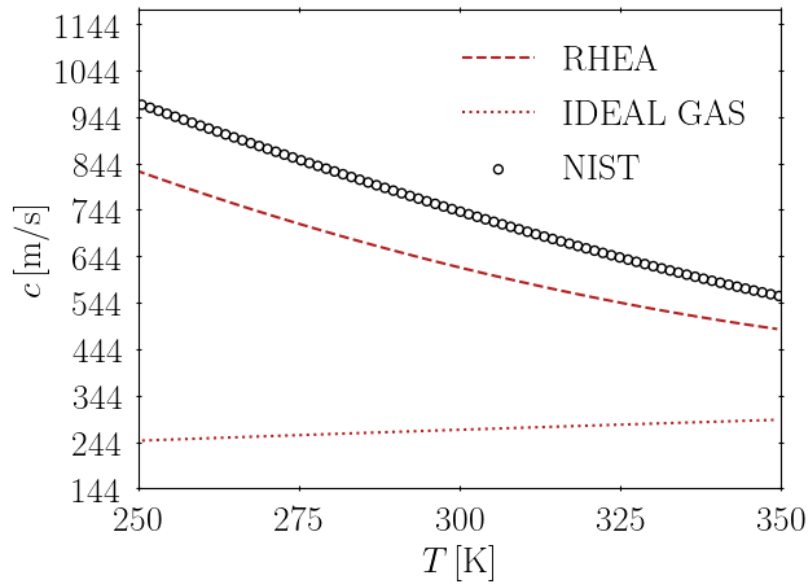
Density vs Temperature



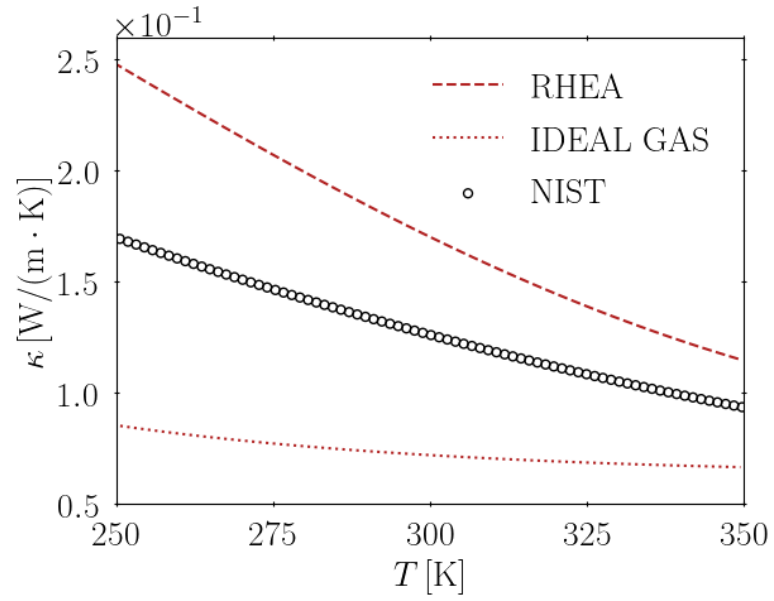
Internal Energy vs Temperature



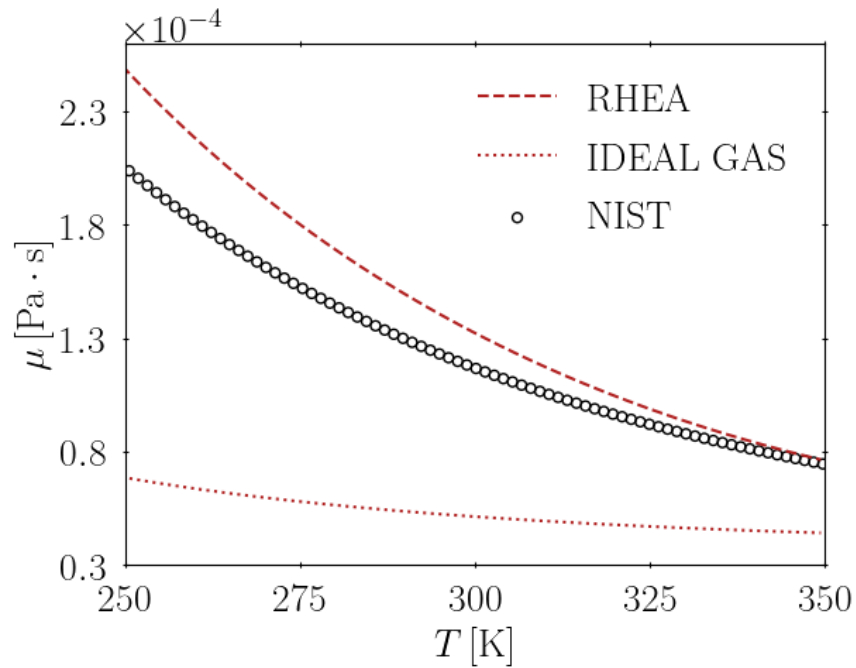
Sound speed vs Temperature



Thermal conductivity vs temperature

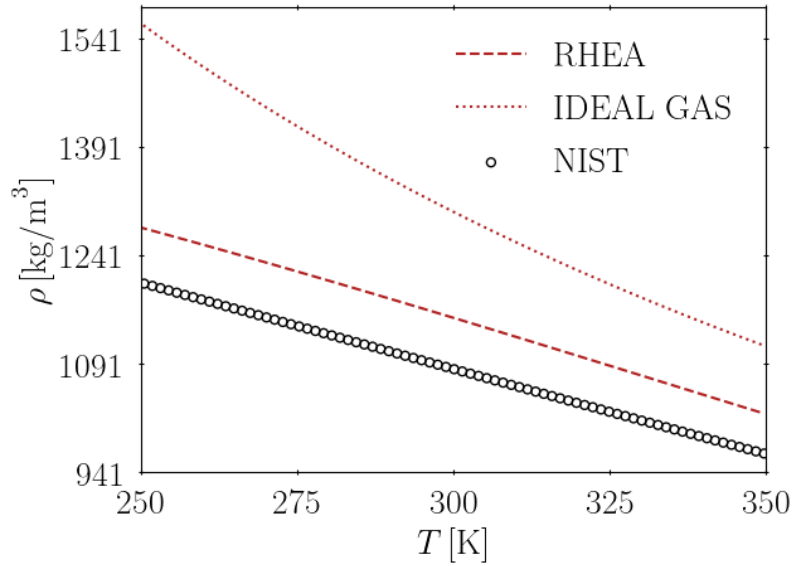


Viscosity vs Temperature

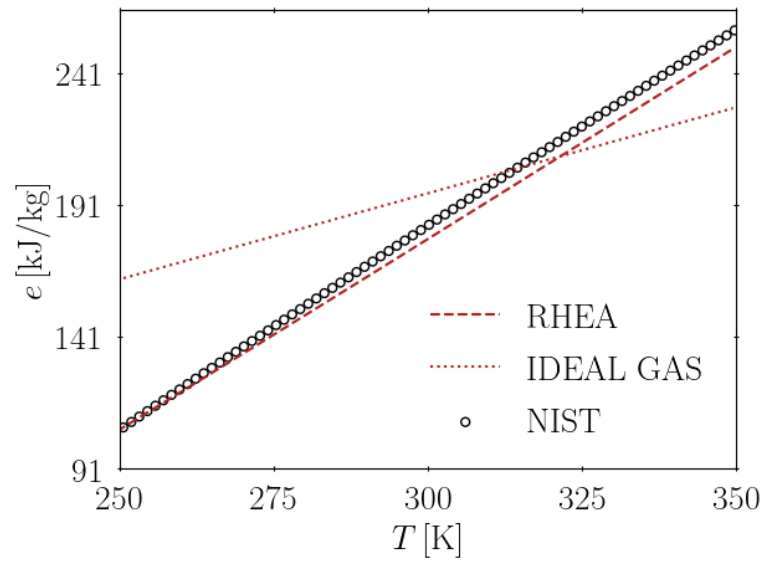


Diòxid de carboni (73,77 MPa)

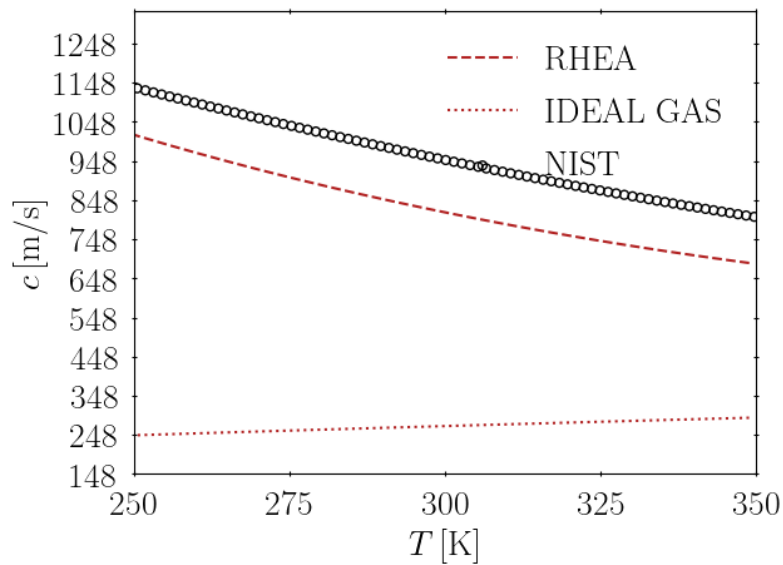
Density vs Temperature



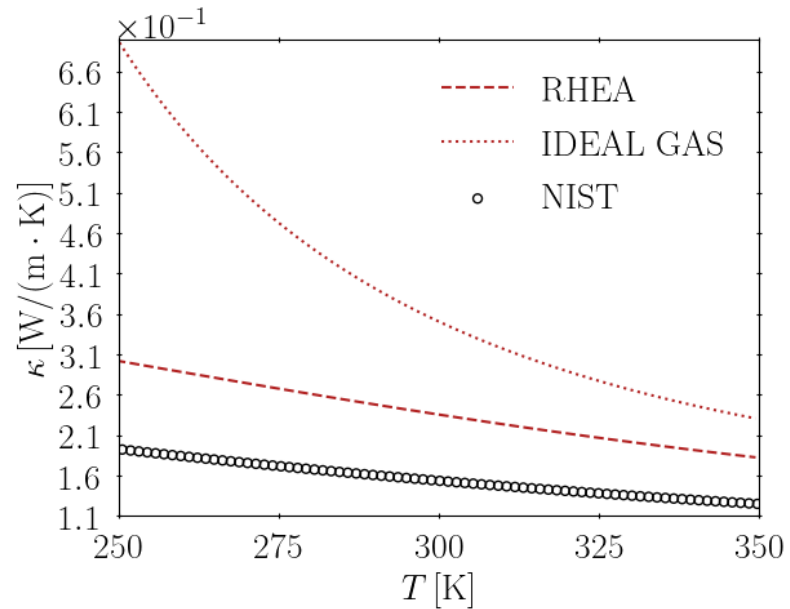
Internal Energy vs Temperature



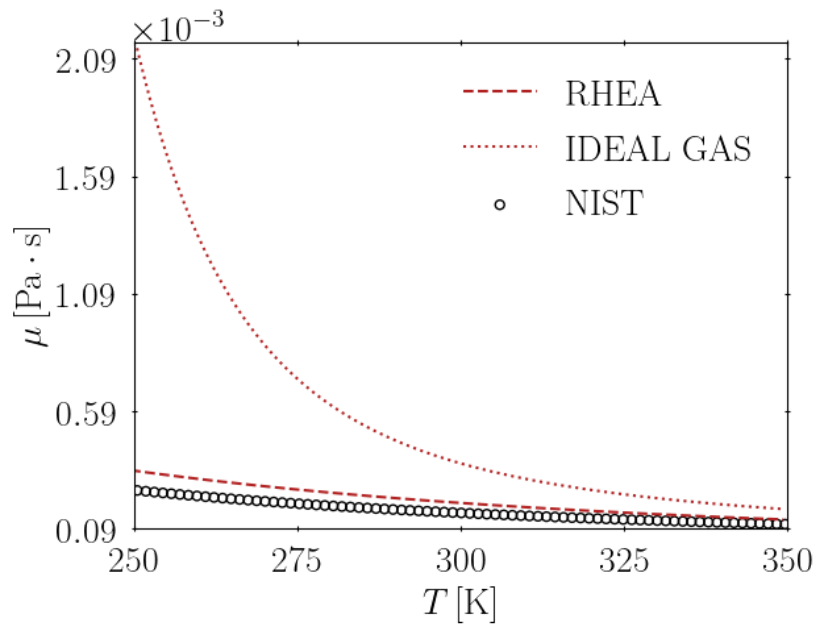
Sound speed vs Temperature



Thermal conductivity vs temperature

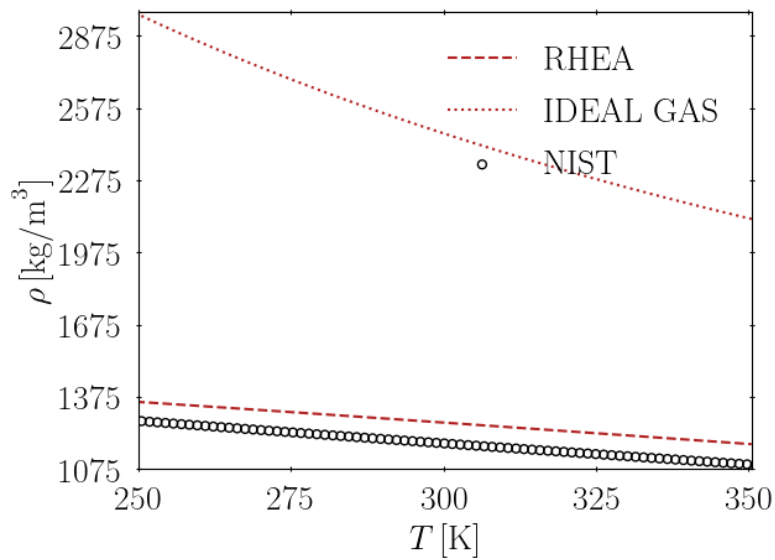


Viscosity vs Temperature

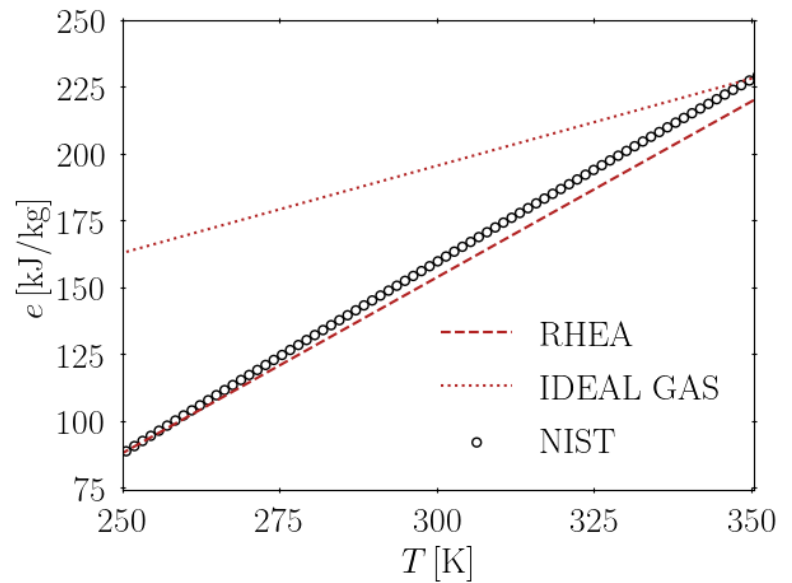


Diòxid de carboni (140 MPa)

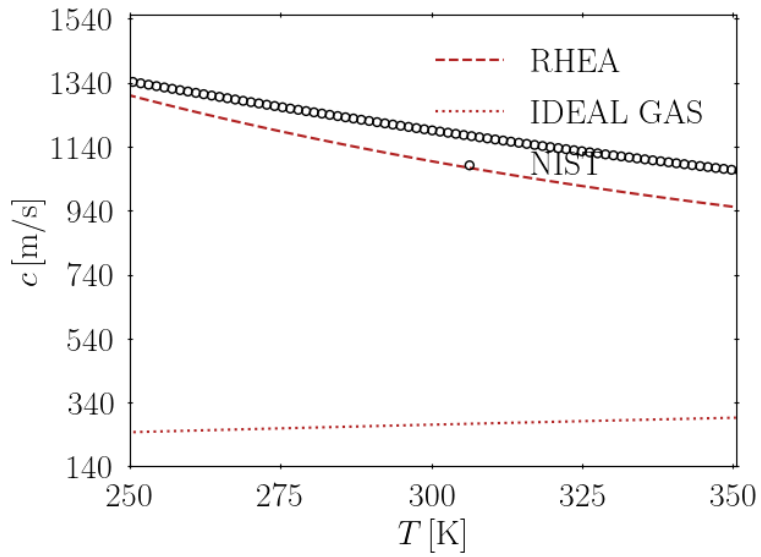
Density vs Temperature



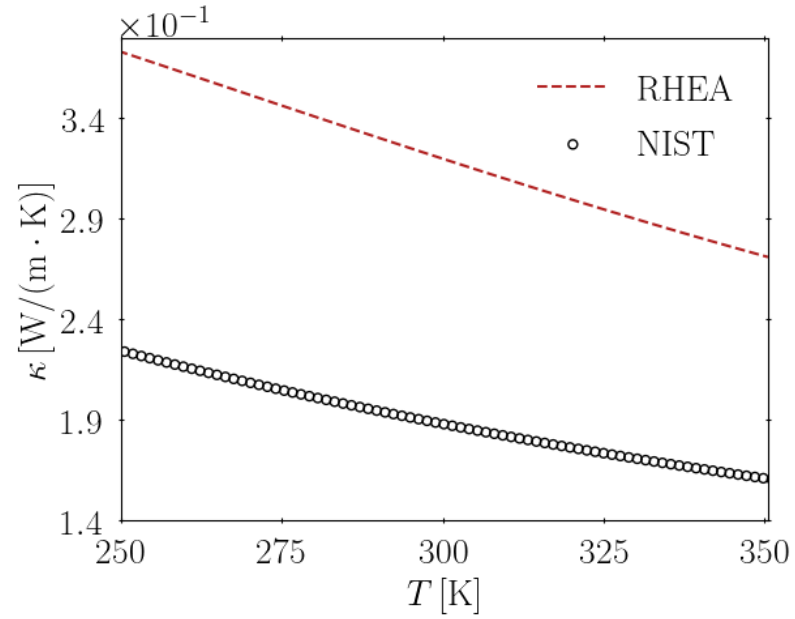
Internal Energy vs Temperature



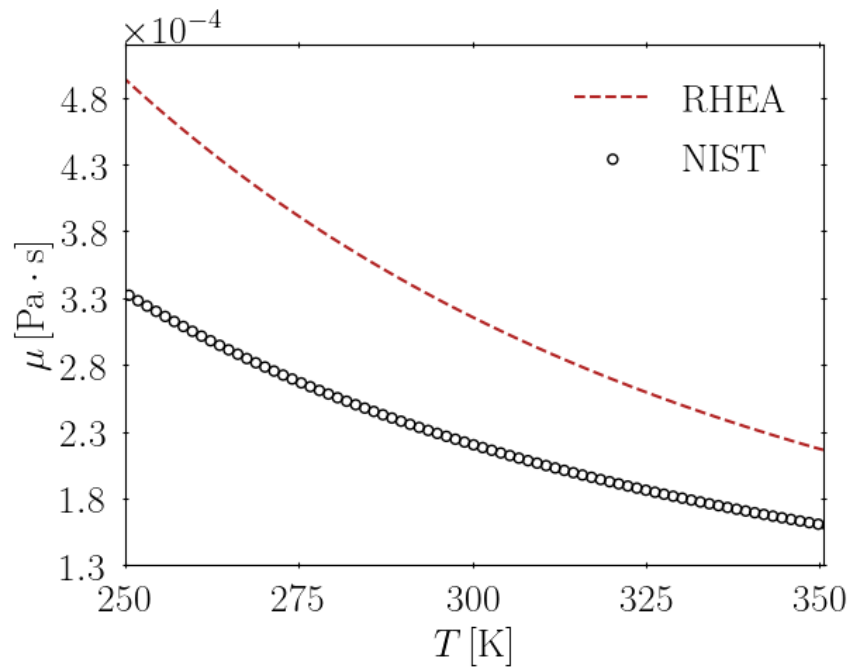
Sound speed vs Temperature



Thermal conductivity vs temperature

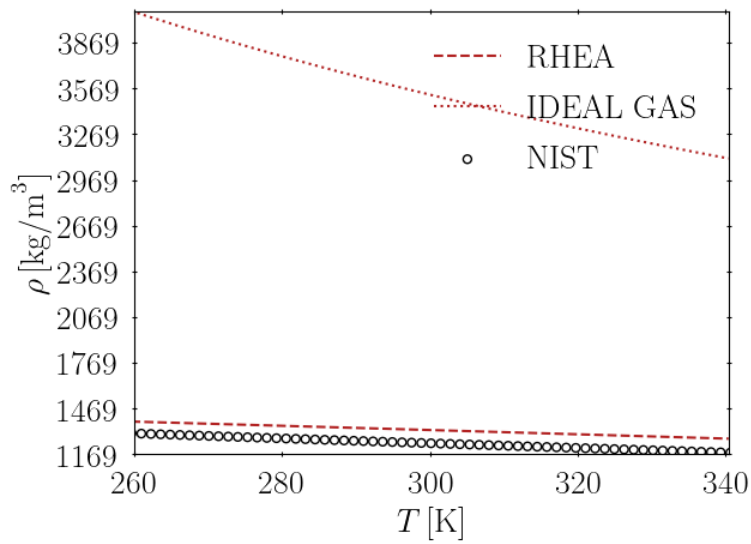


Viscosity vs Temperature

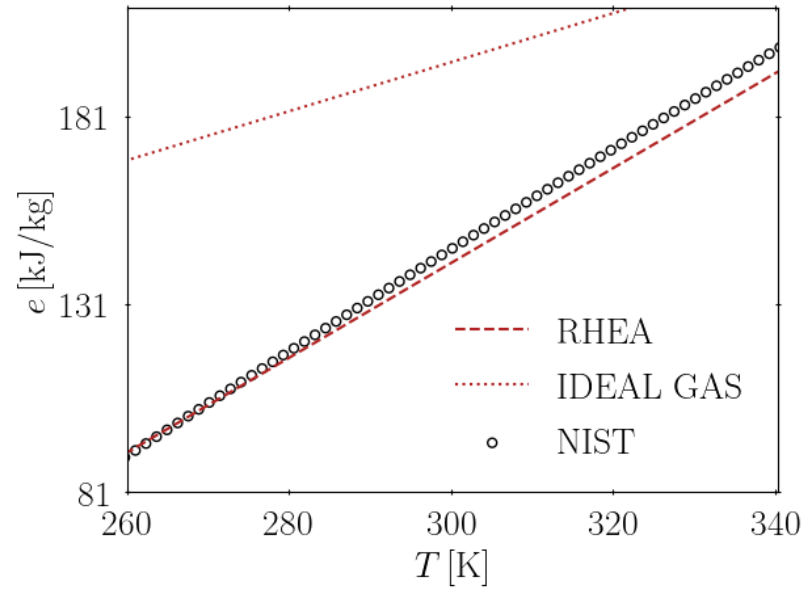


Diòxid de carboni (200 MPa)

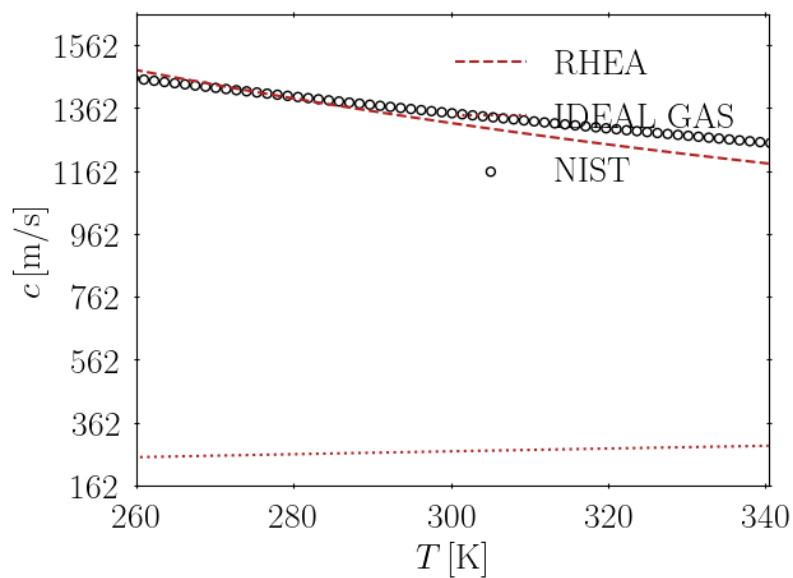
Density vs Temperature



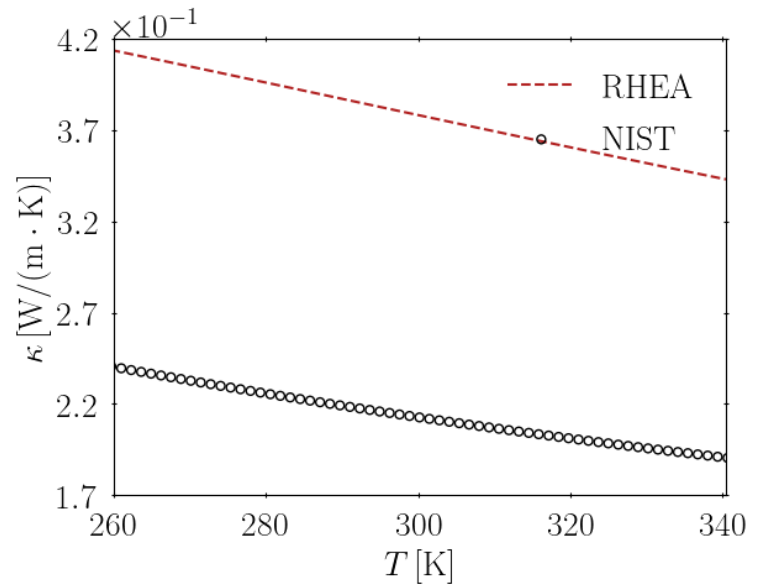
Internal Energy vs Temperature



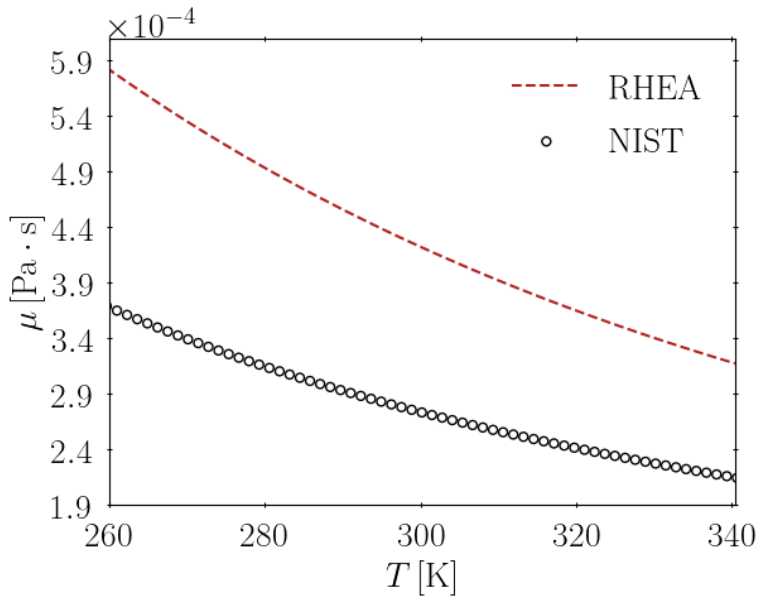
Sound speed vs Temperature



Thermal conductivity vs temperature

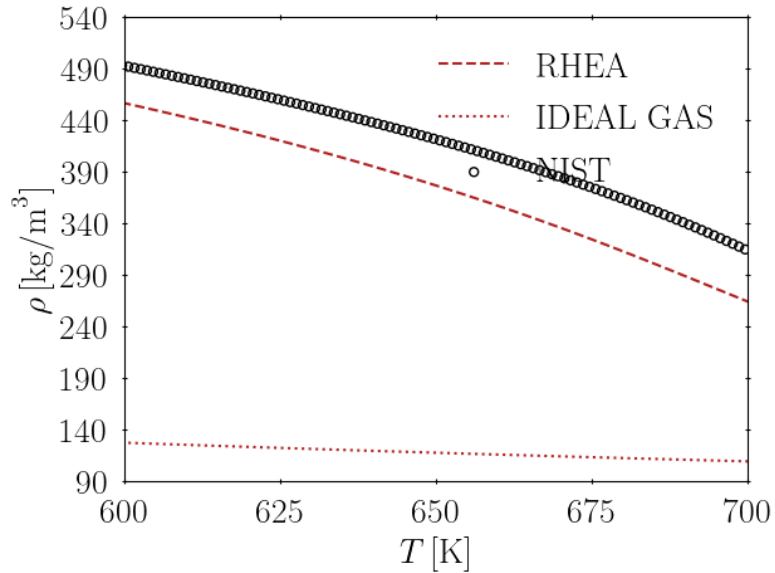


Viscosity vs Temperature

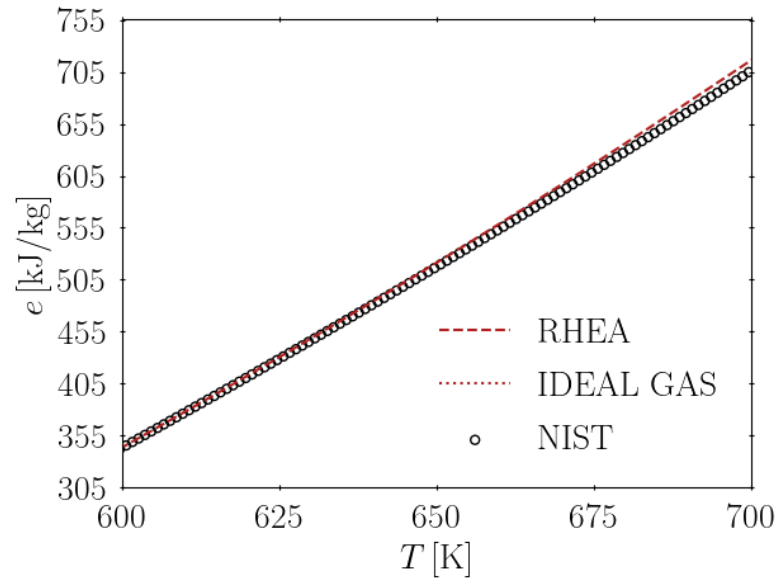


Dodecà 3,63 MPa

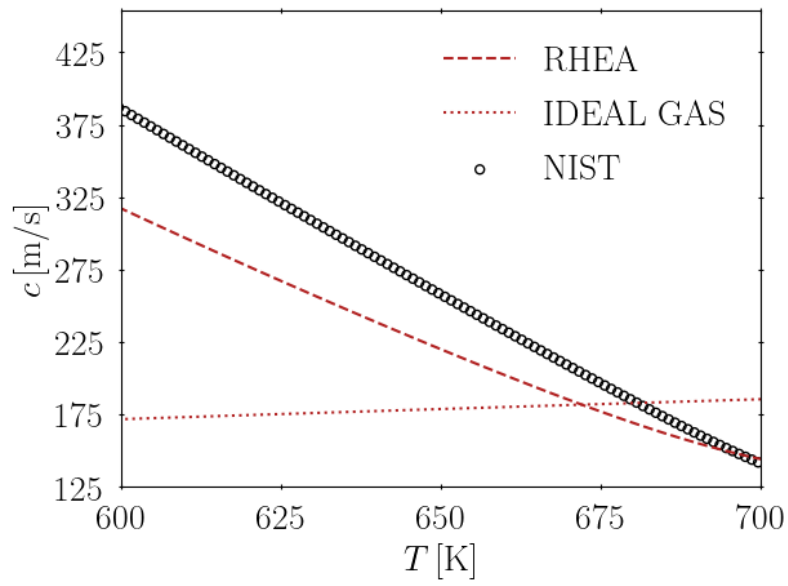
Density vs Temperature



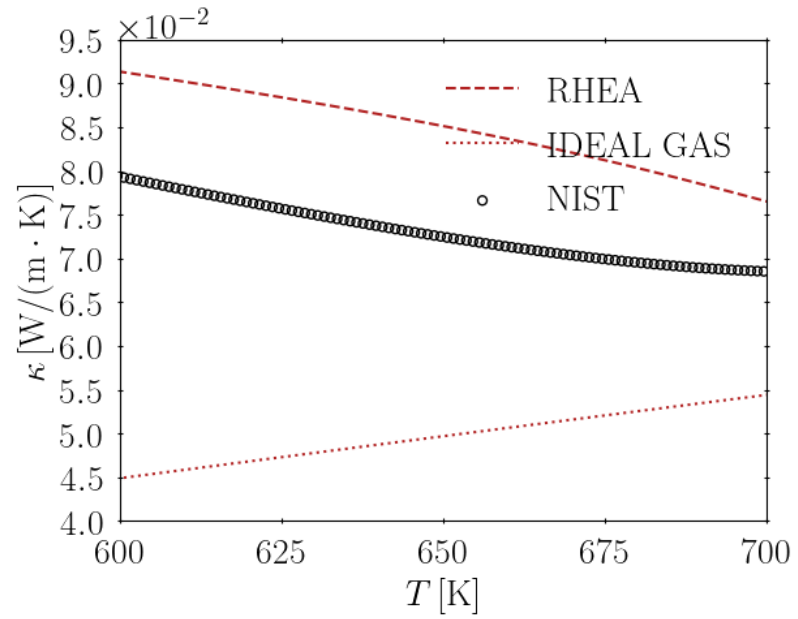
Internal Energy vs Temperature



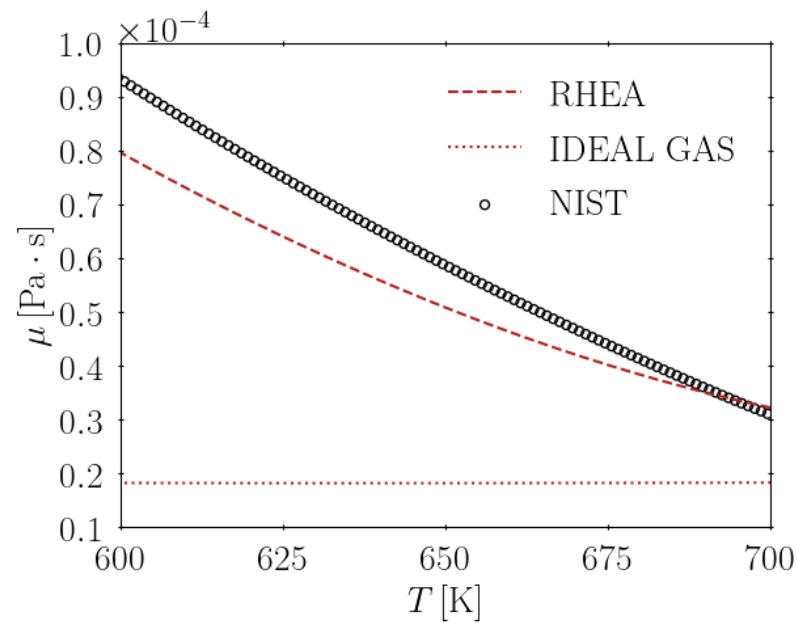
Sound speed vs Temperature



Thermal conductivity vs temperature

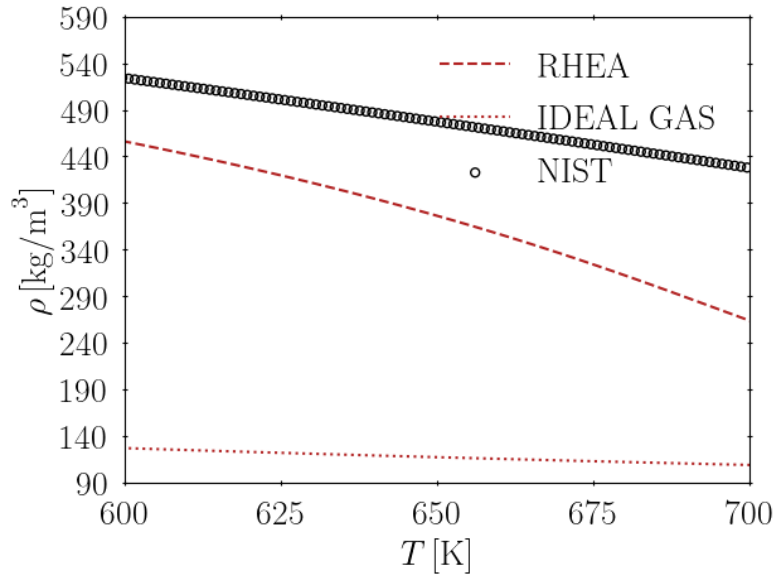


Viscosity vs Temperature

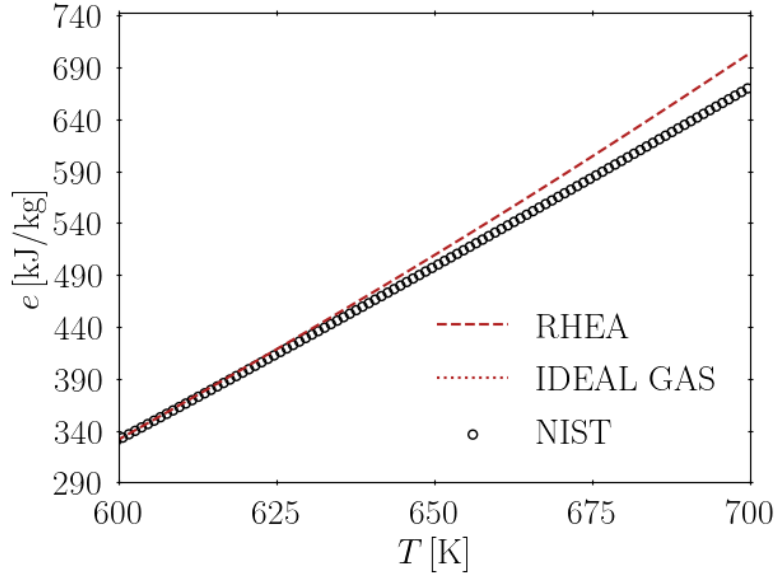


Dodecà 9,09 MPa

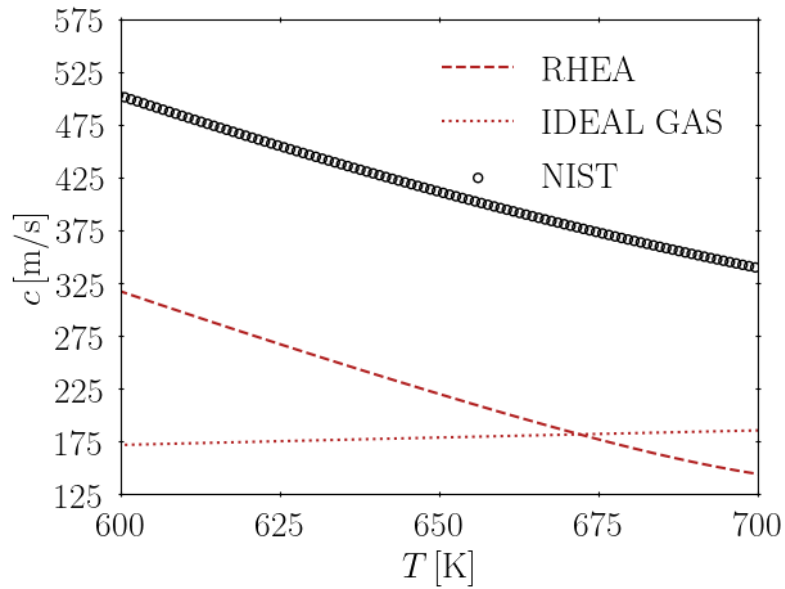
Density vs Temperature



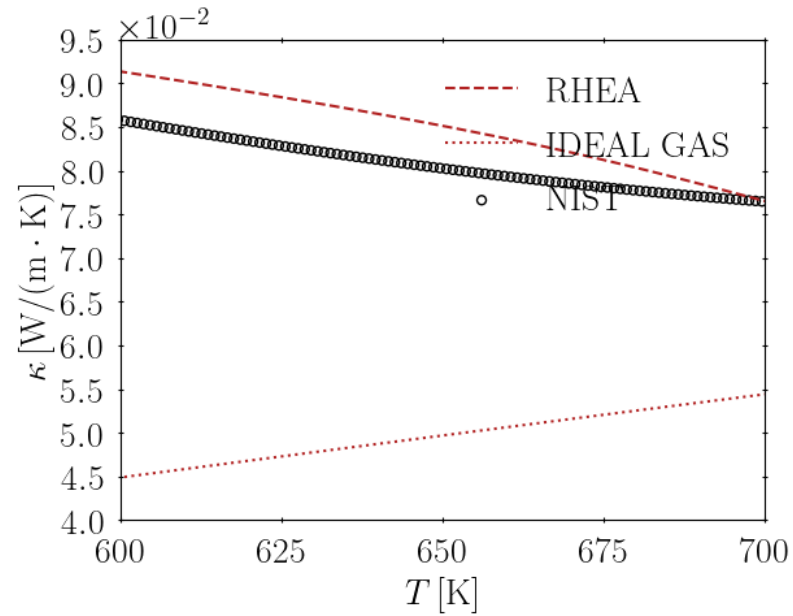
Internal Energy vs Temperature



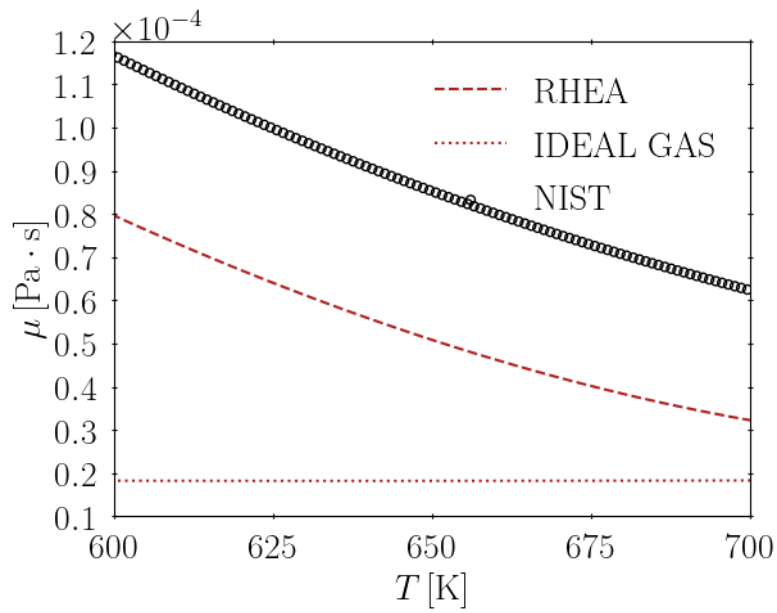
Sound speed vs Temperature



Thermal conductivity vs temperature

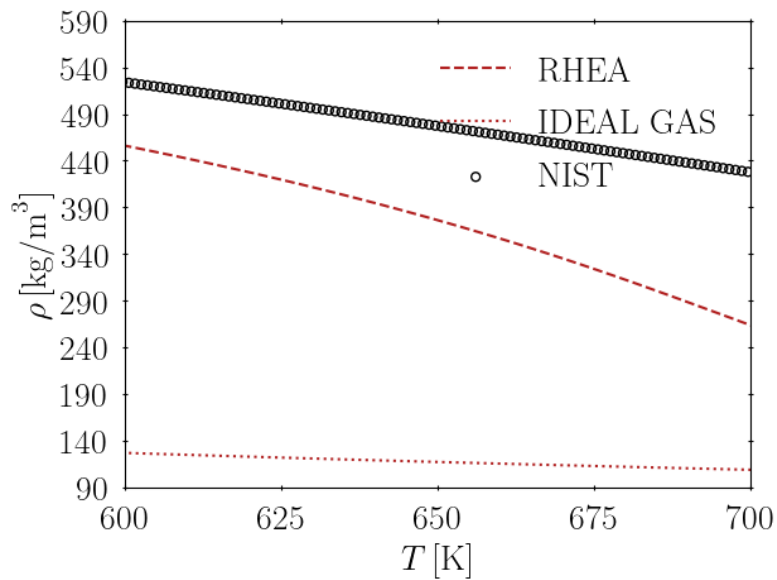


Viscosity vs Temperature

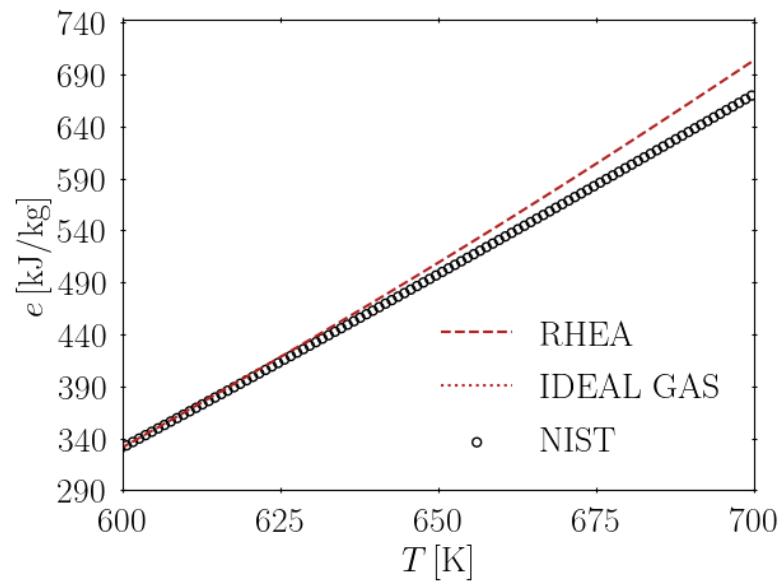


Dodecà 18,17 MPa

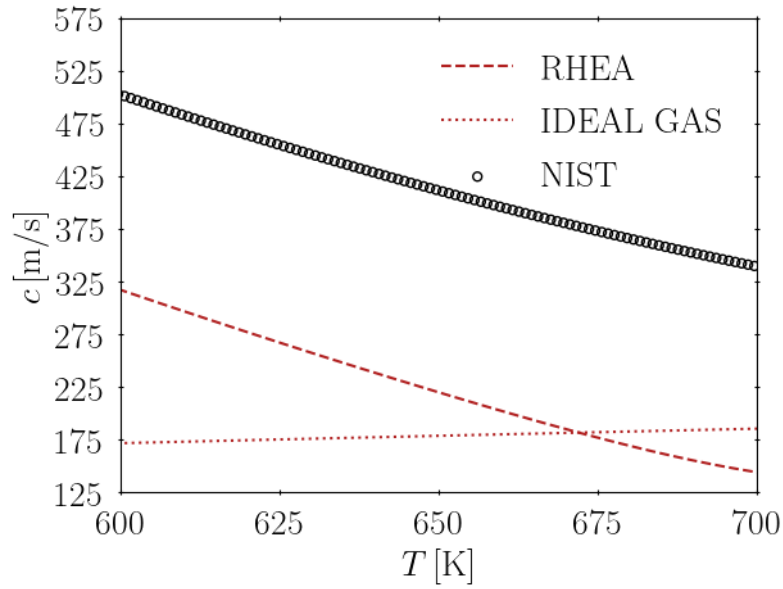
Density vs Temperature



Internal Energy vs Temperature



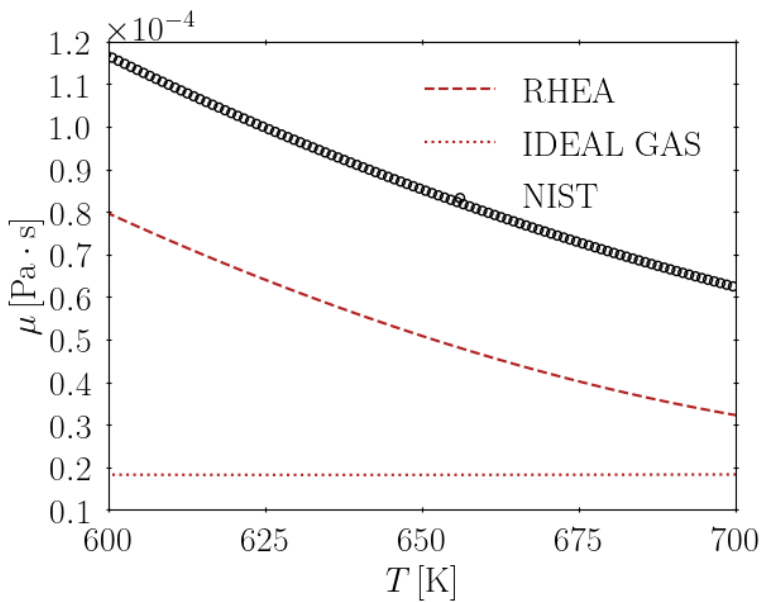
Sound speed vs Temperature



Thermal conductivity vs temperature

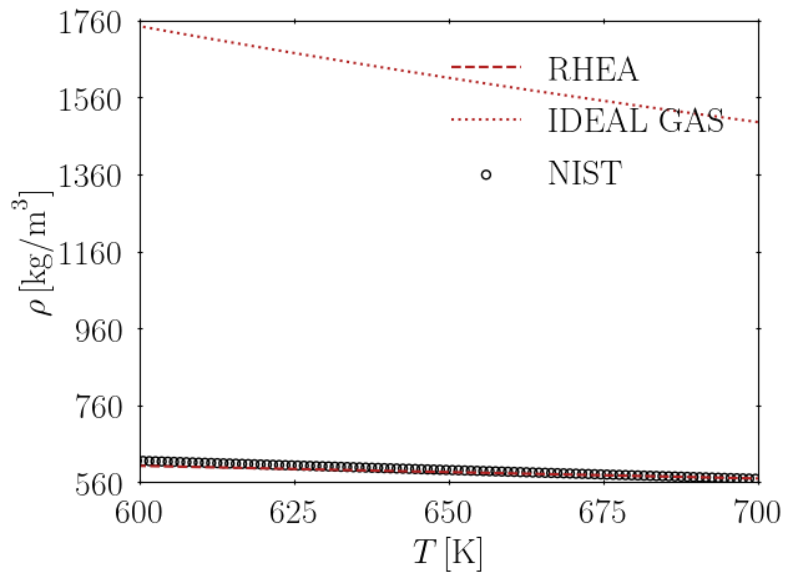


Viscosity vs Temperature

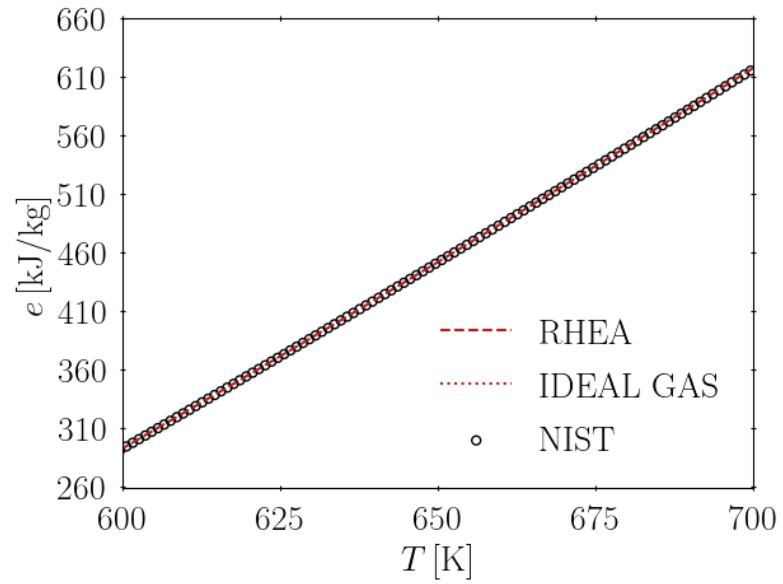


Dodecà 50 MPa

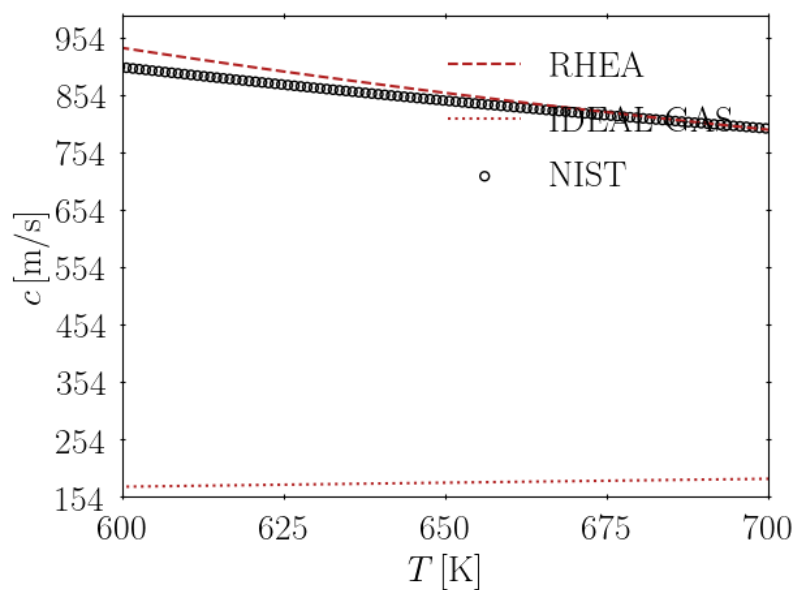
Density vs Temperature



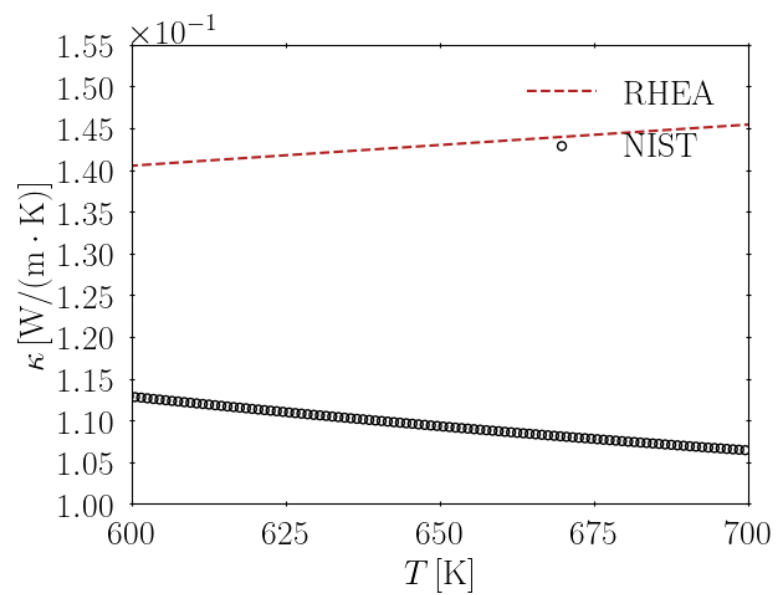
Internal Energy vs Temperature



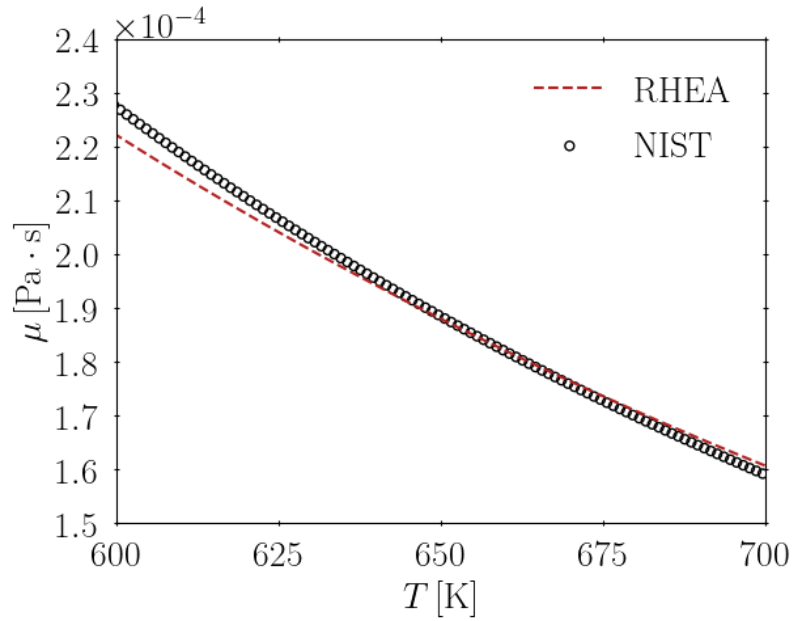
Sound speed vs Temperature



Thermal conductivity vs temperature

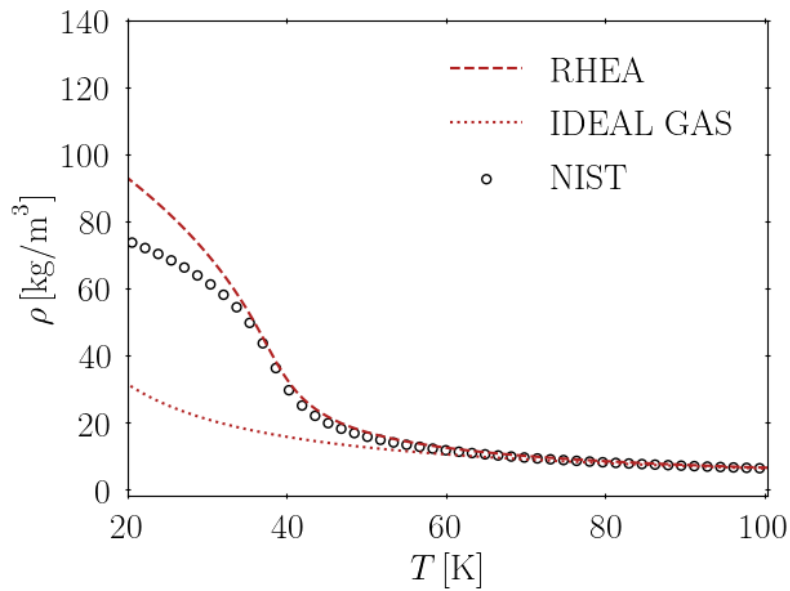


Viscosity vs Temperature

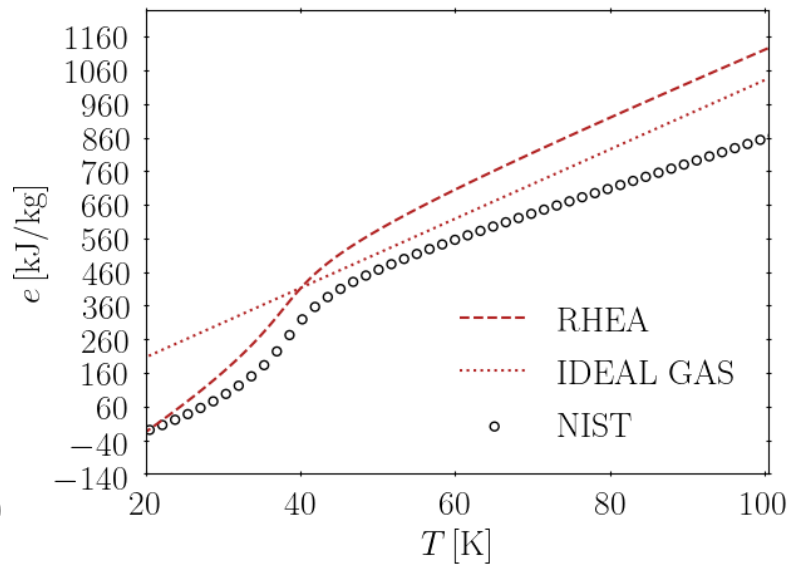


Hidrogen (2,59 MPa)

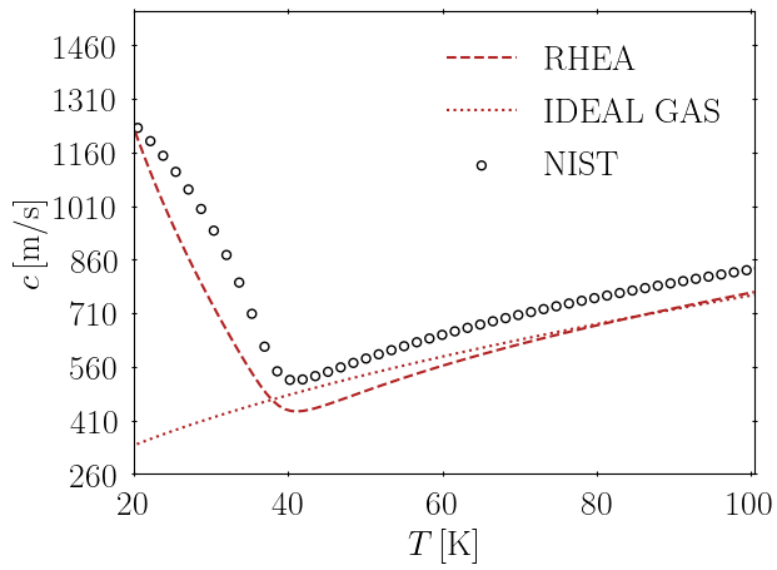
Density vs Temperature



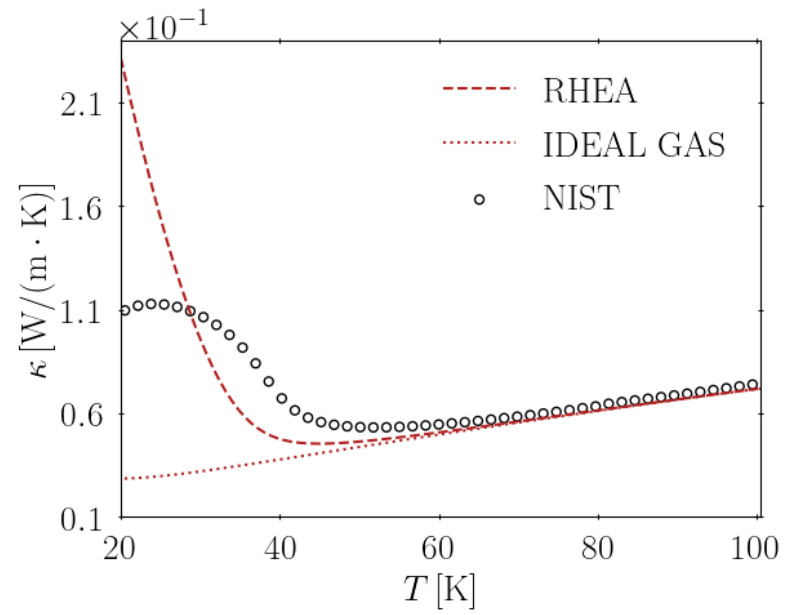
Internal Energy vs Temperature



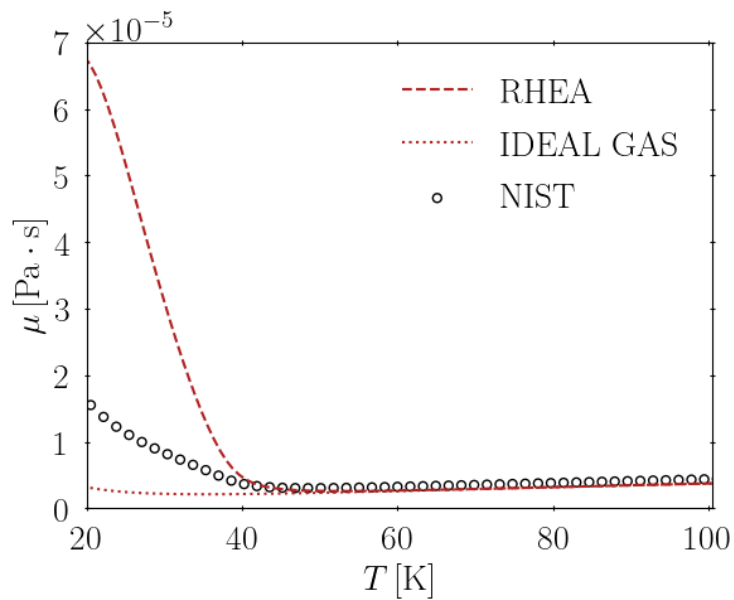
Sound speed vs Temperature



Thermal conductivity vs temperature

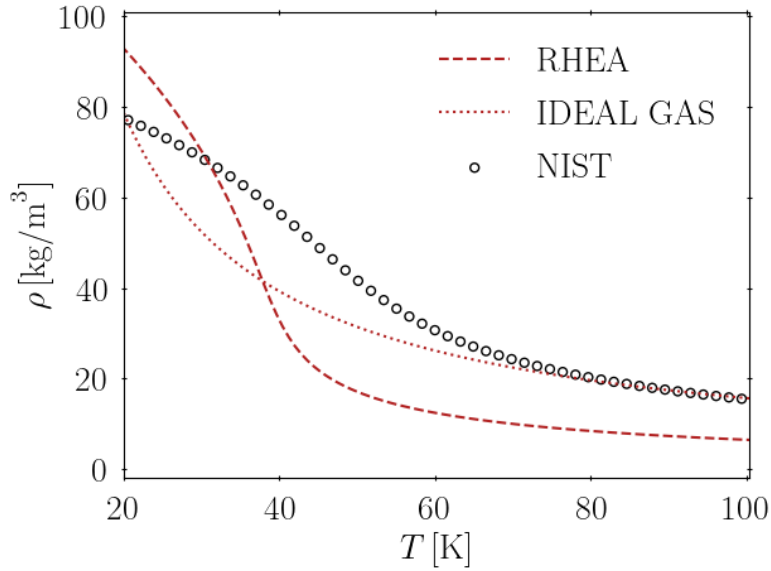


Viscosity vs Temperature

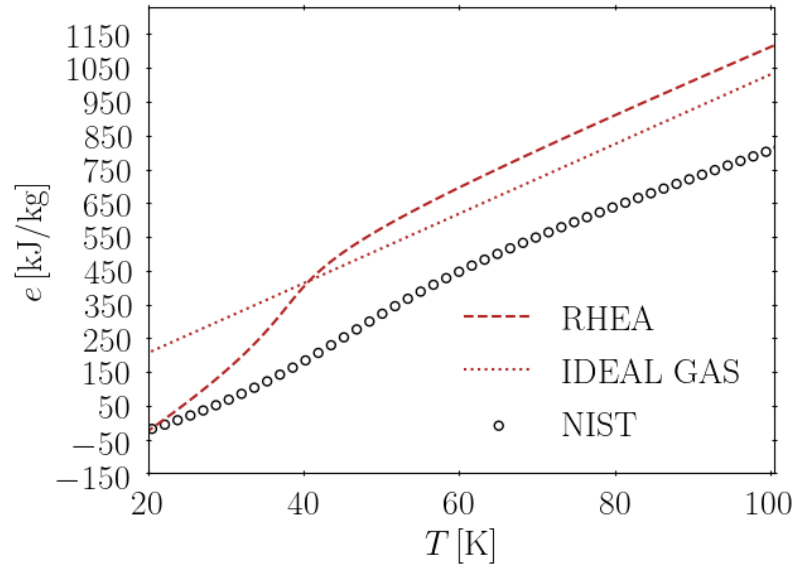


Hidrogen (6,48 MPa)

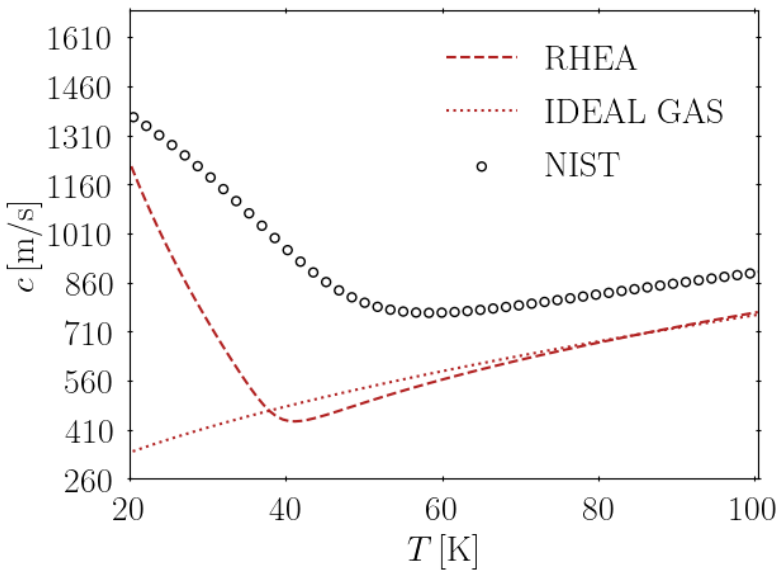
Density vs Temperature



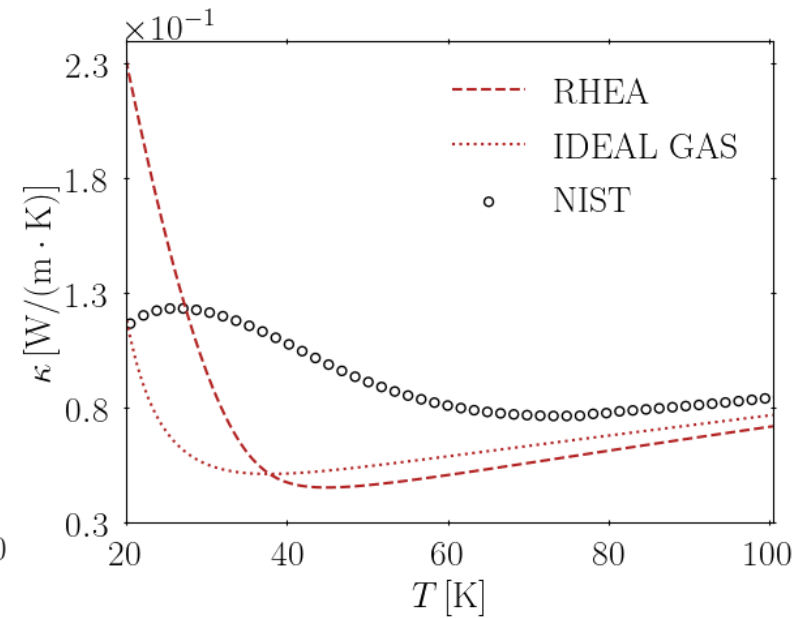
Internal Energy vs Temperature



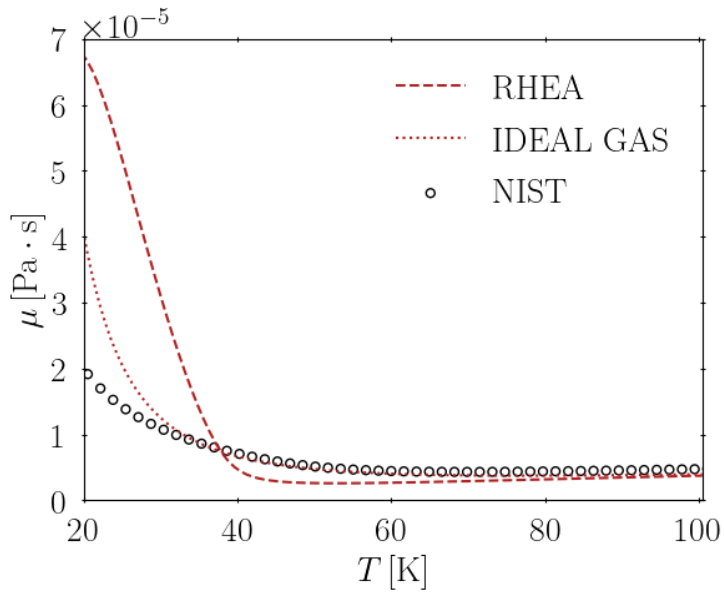
Sound speed vs Temperature



Thermal conductivity vs temperature

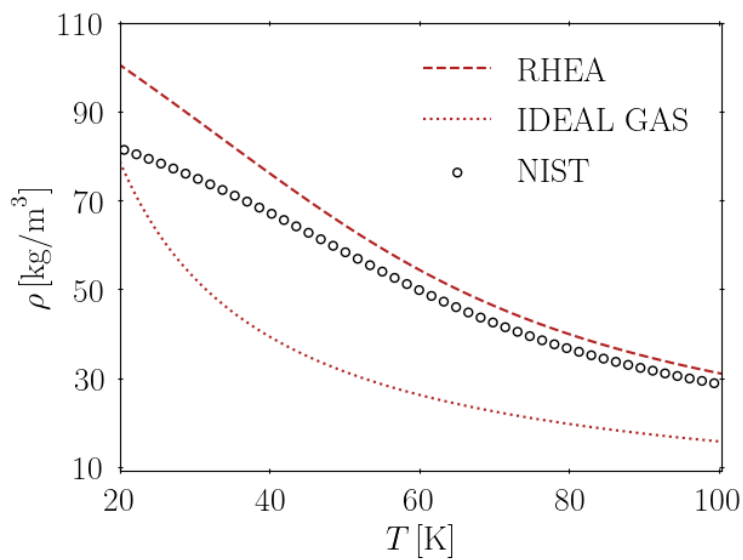


Viscosity vs Temperature

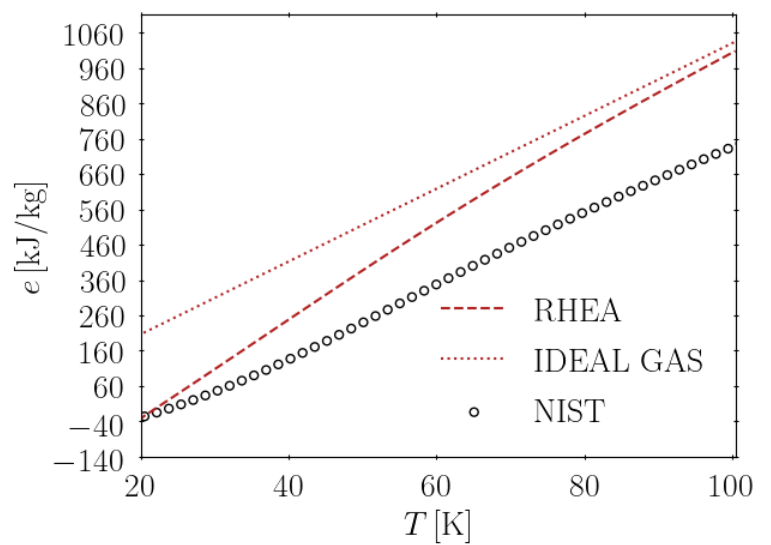


Hydrogen (12,96 MPa)

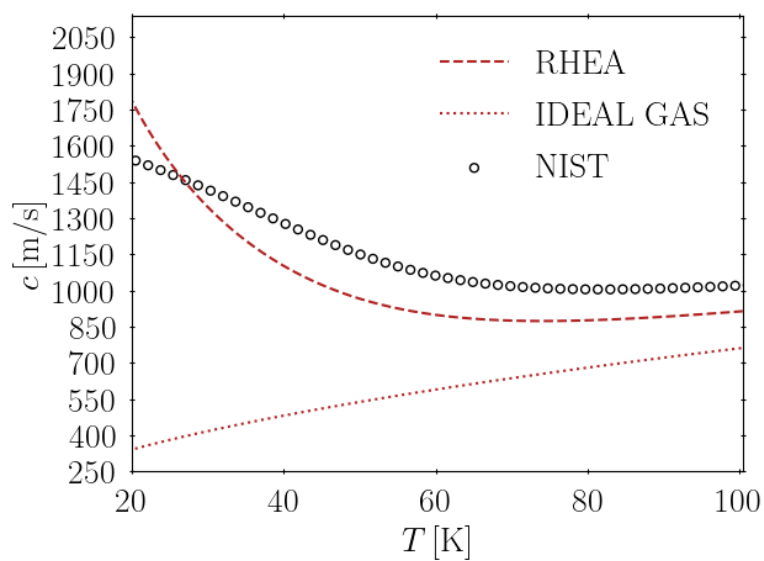
Density vs Temperature



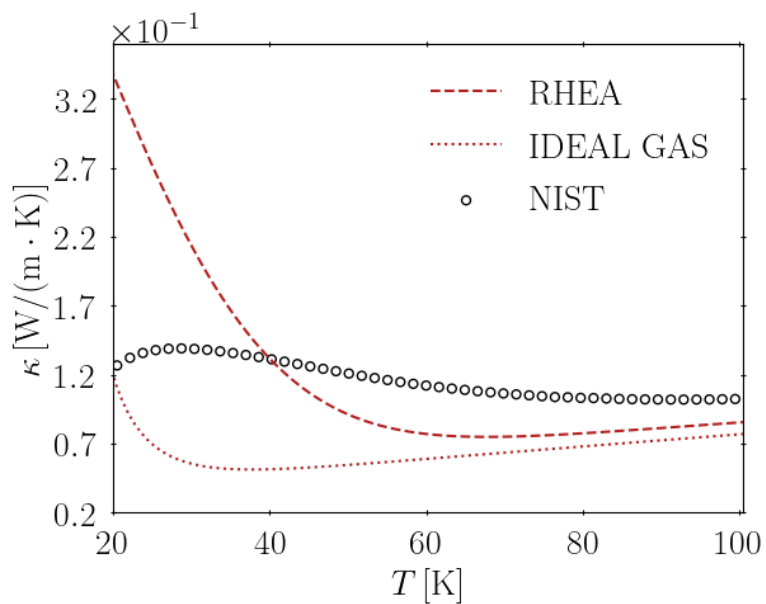
Internal Energy vs Temperature



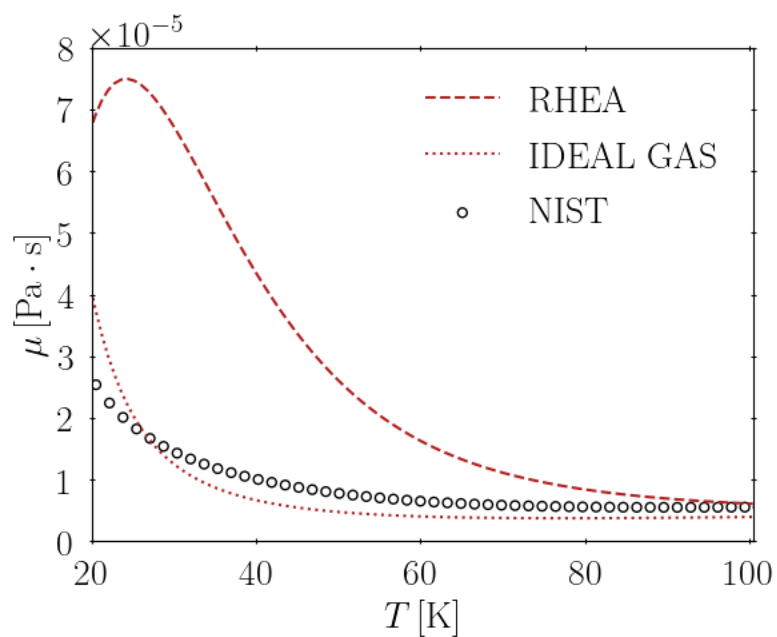
Sound speed vs Temperature



Thermal conductivity vs temperature

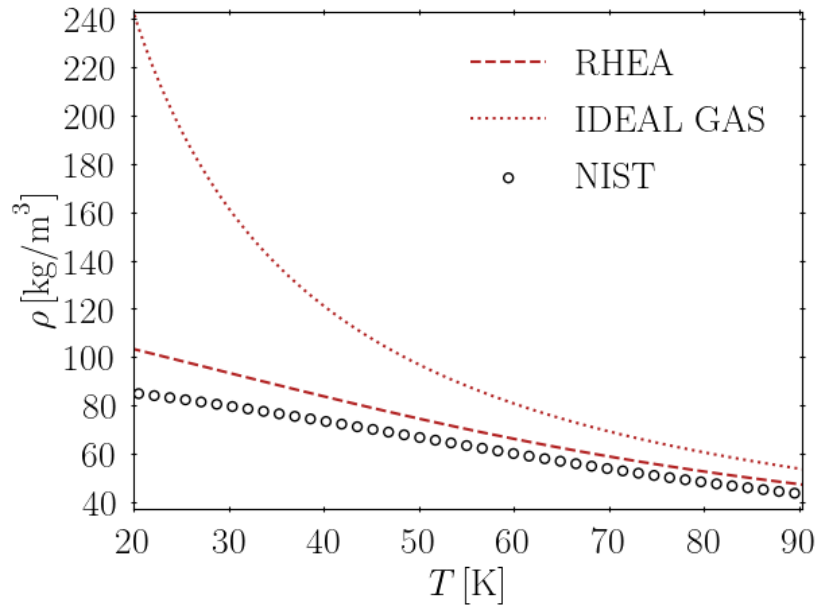


Viscosity vs Temperature

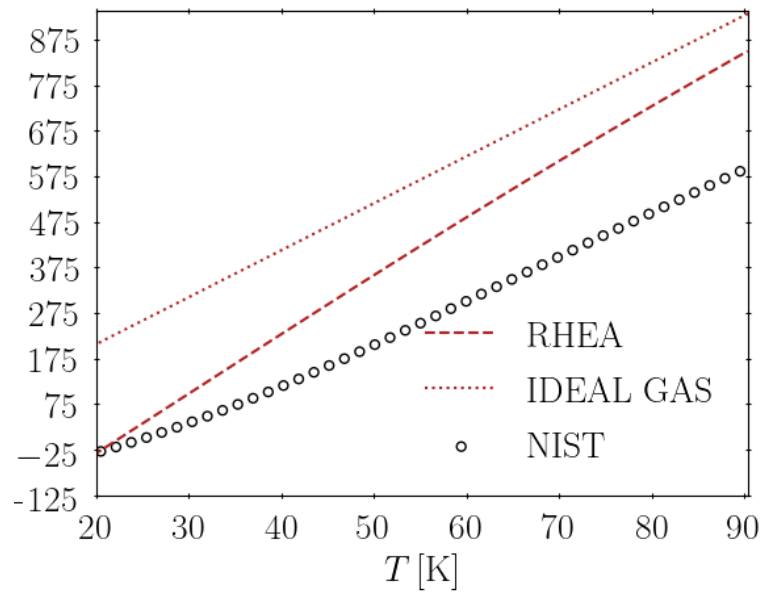


Hidrogen (20 MPa)

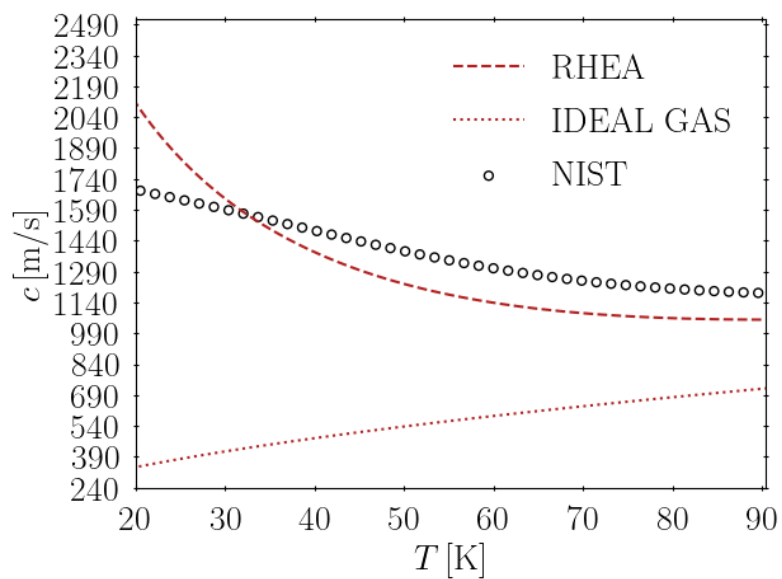
Density vs Temperature



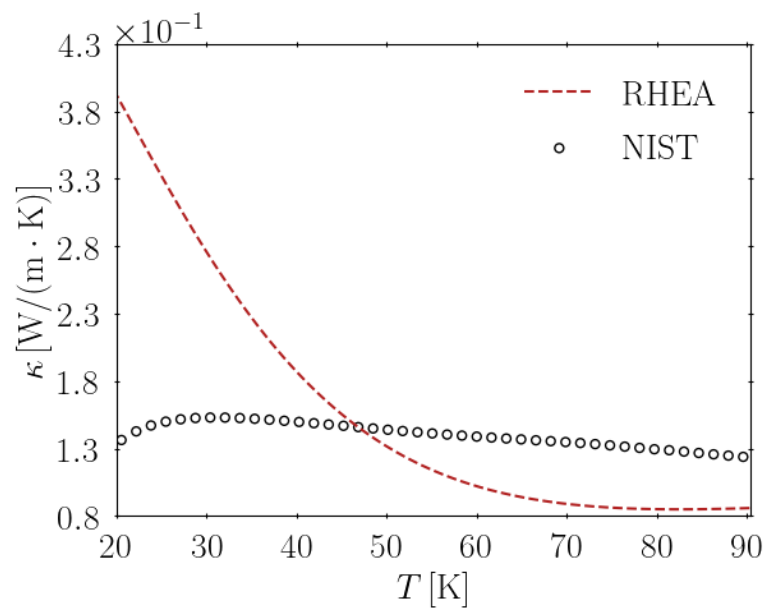
Internal Energy vs Temperature



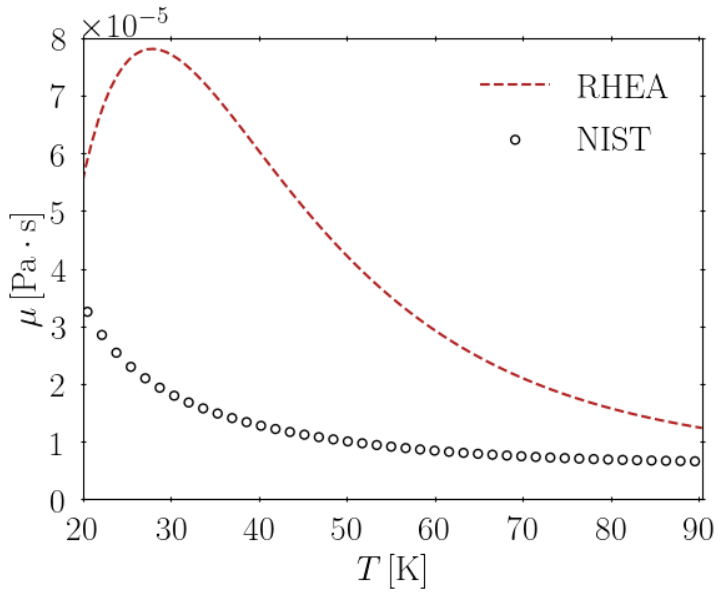
Sound speed vs Temperature



Thermal conductivity vs temperature

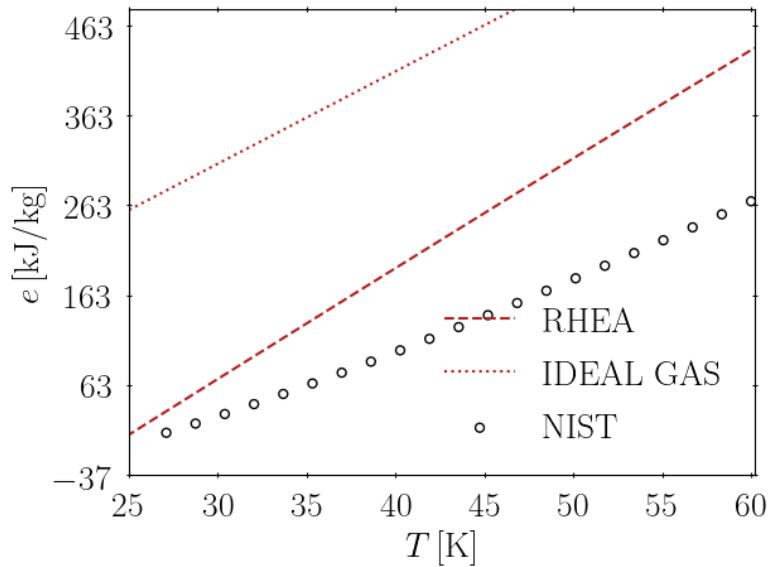


Viscosity vs Temperature

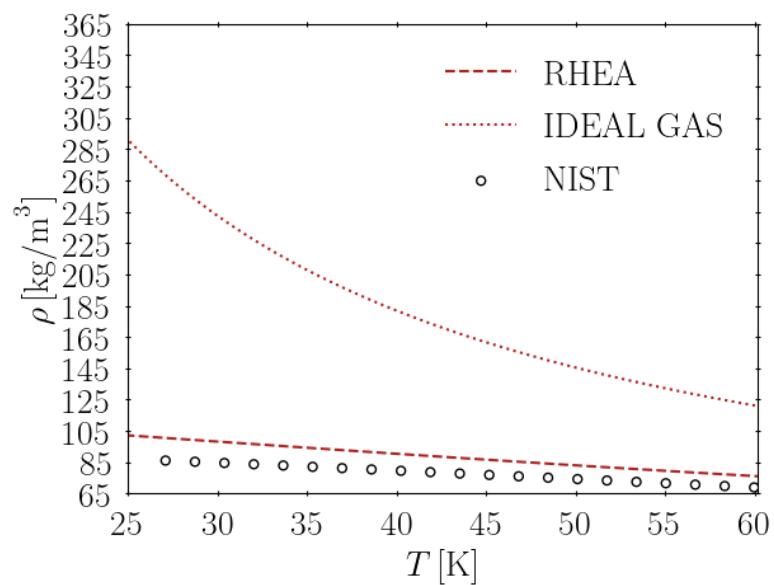


Hidrogen (30 MPa)

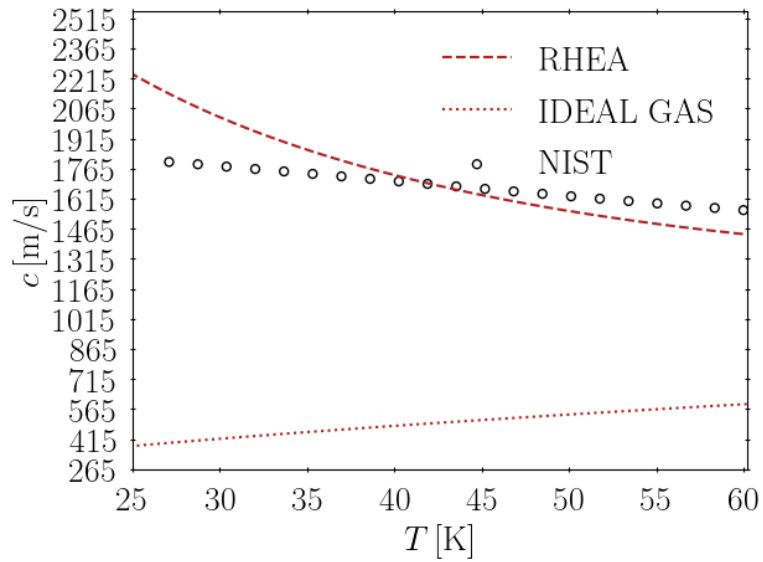
Density vs Temperature



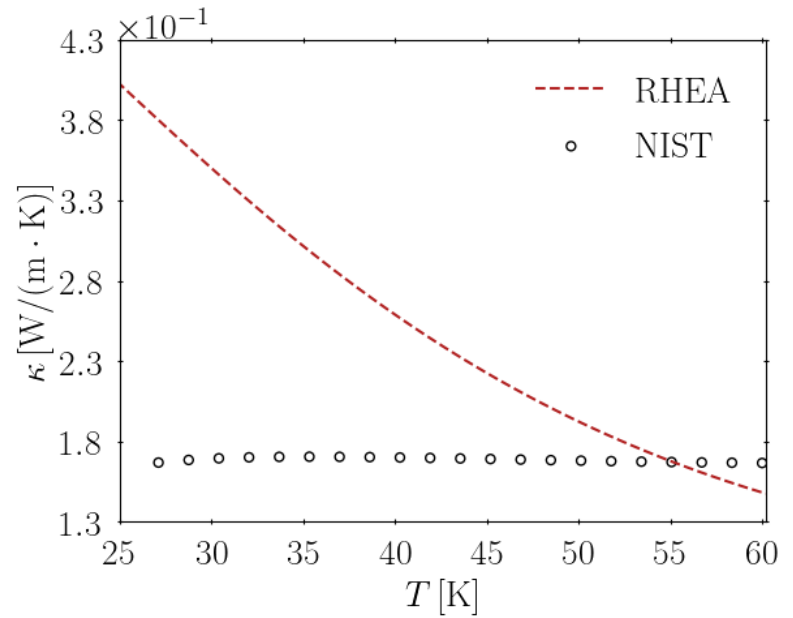
Internal Energy vs Temperature



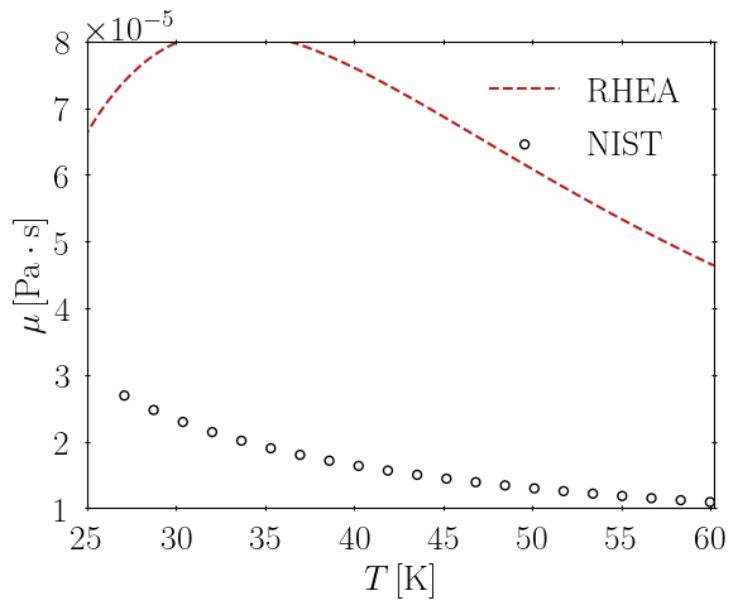
Sound speed vs Temperature



Thermal conductivity vs temperature

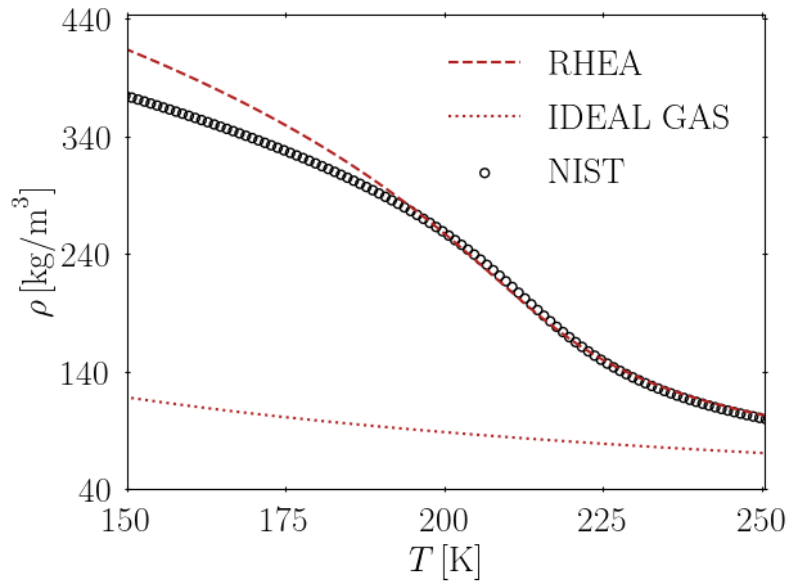


Viscosity vs Temperature

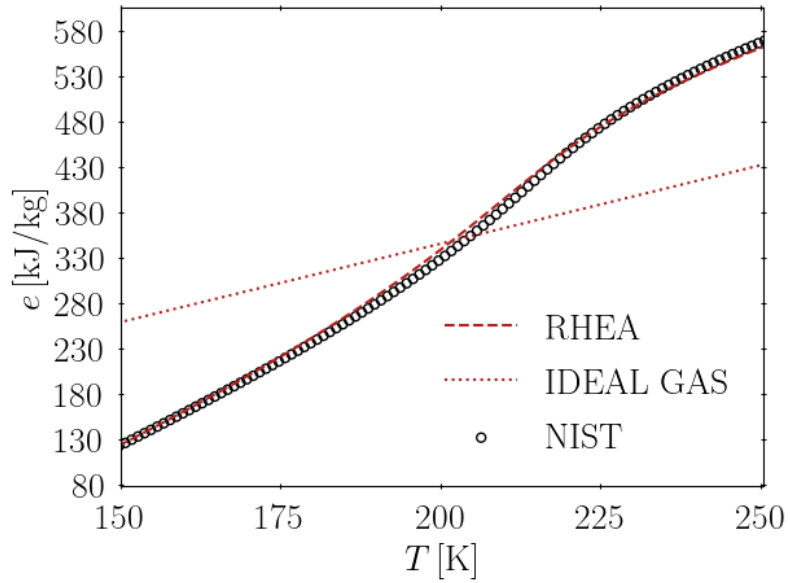


Metà (9,19 MPa)

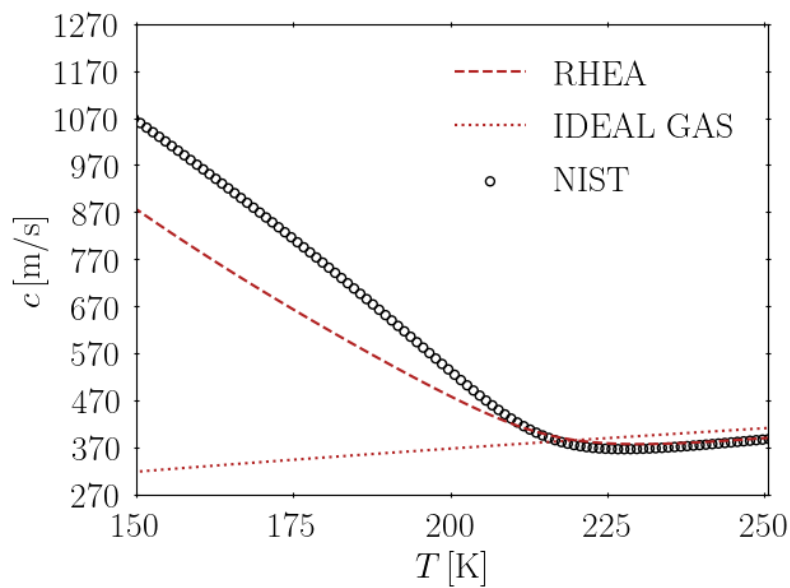
Density vs Temperature



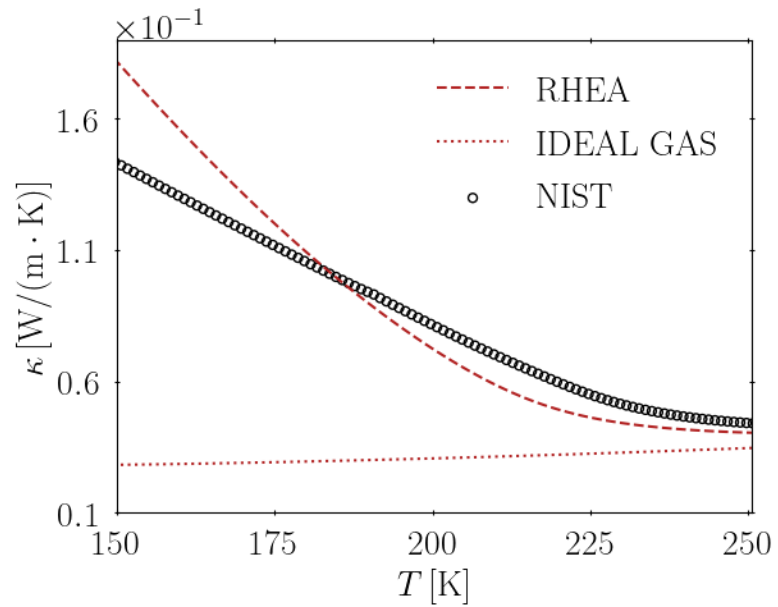
Internal Energy vs Temperature



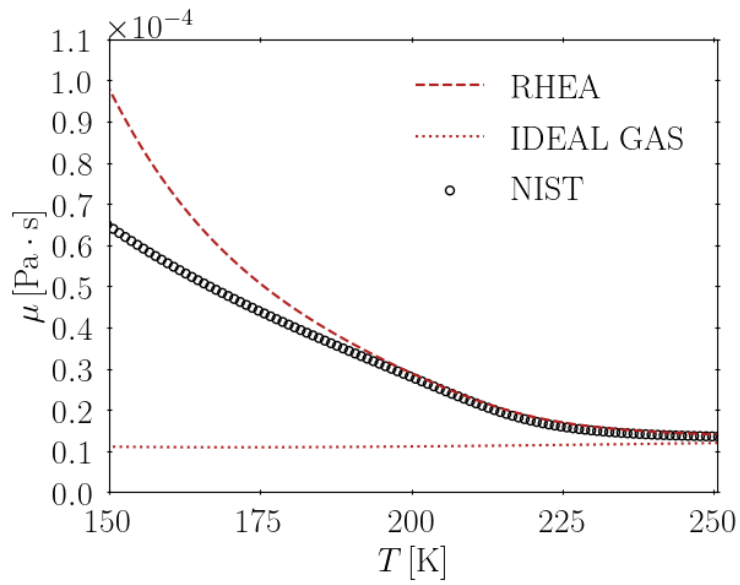
Sound speed vs Temperature



Thermal conductivity vs temperature

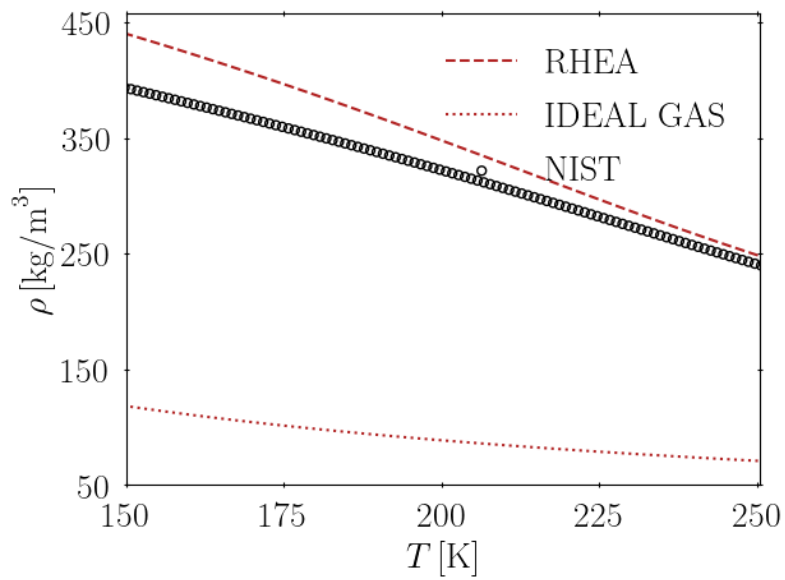


Viscosity vs Temperature

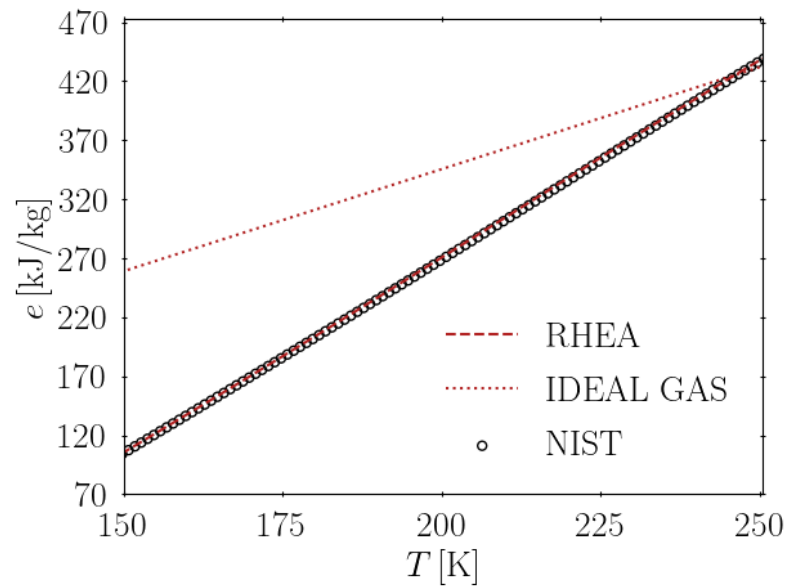


Metà (22,99 MPa)

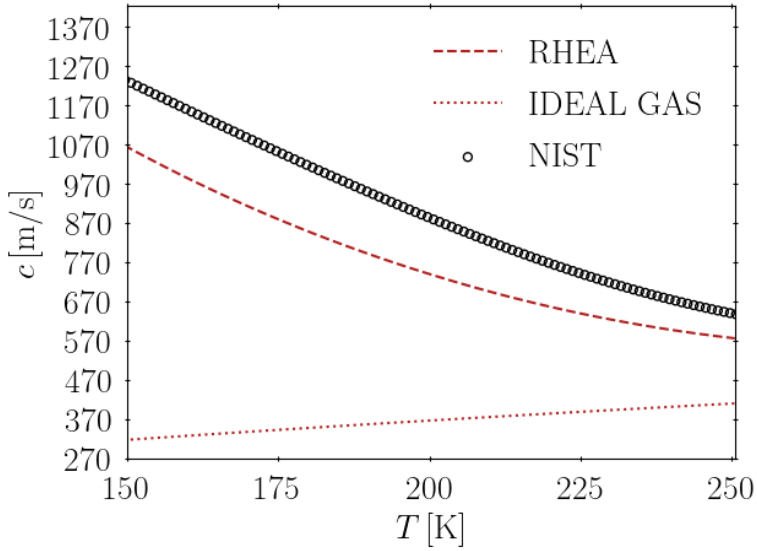
Density vs Temperature



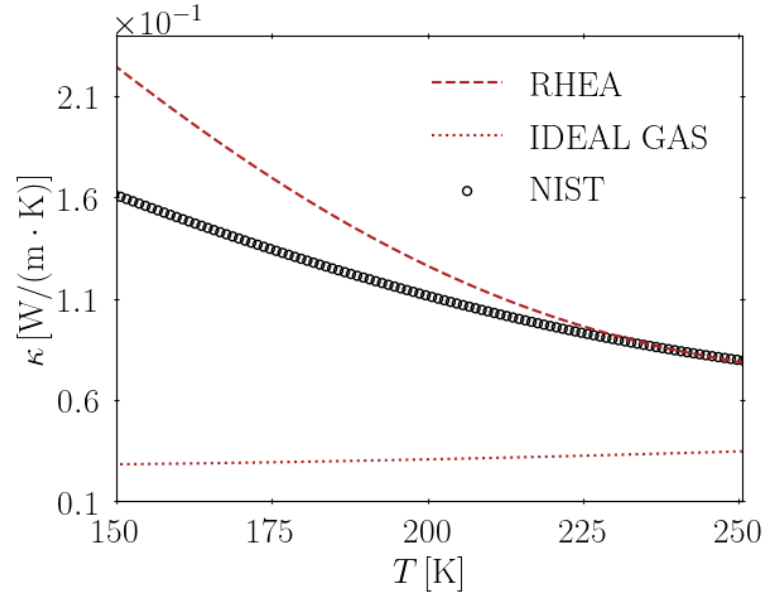
Internal Energy vs Temperature



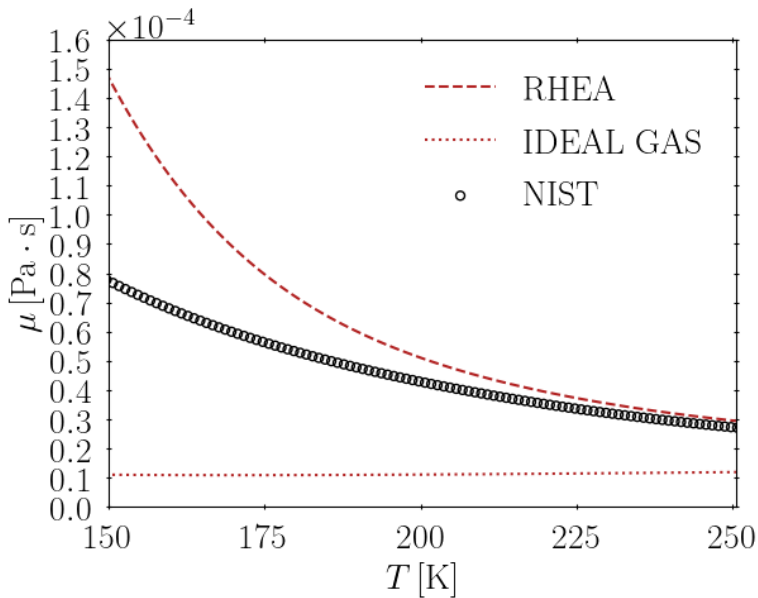
Sound speed vs Temperature



Thermal conductivity vs temperature

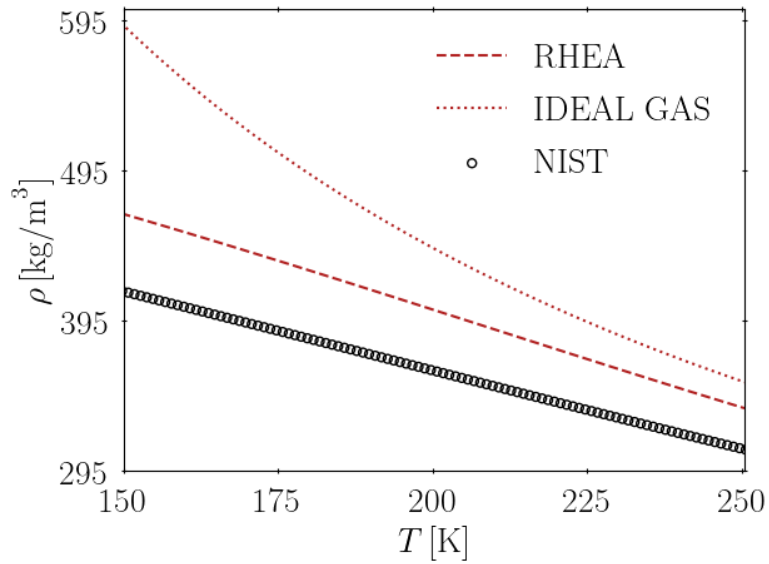


Viscosity vs Temperature

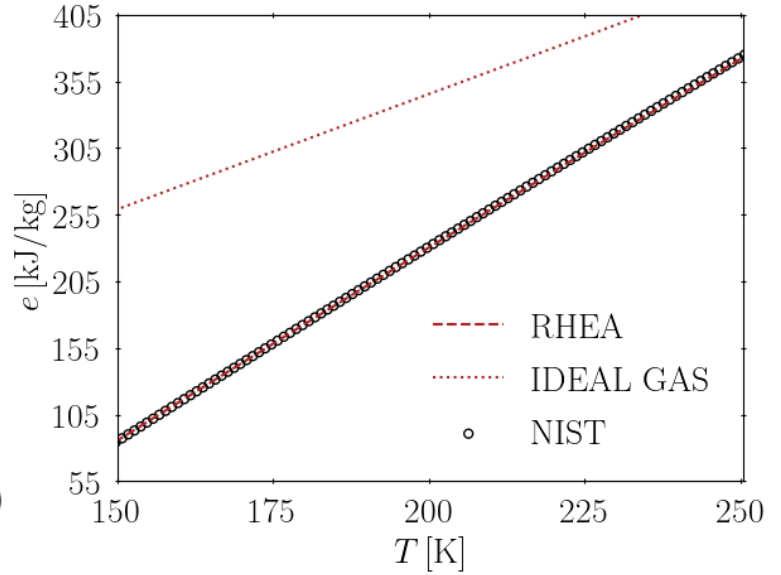


Metà (45,99 MPa)

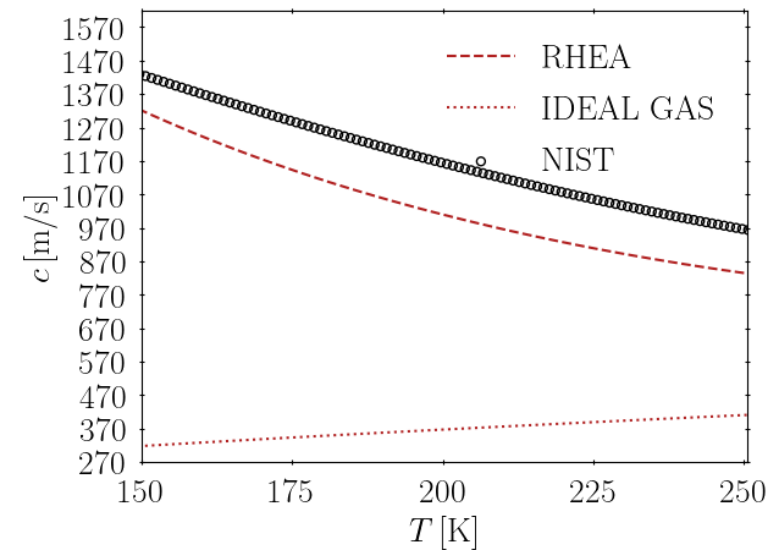
Density vs Temperature



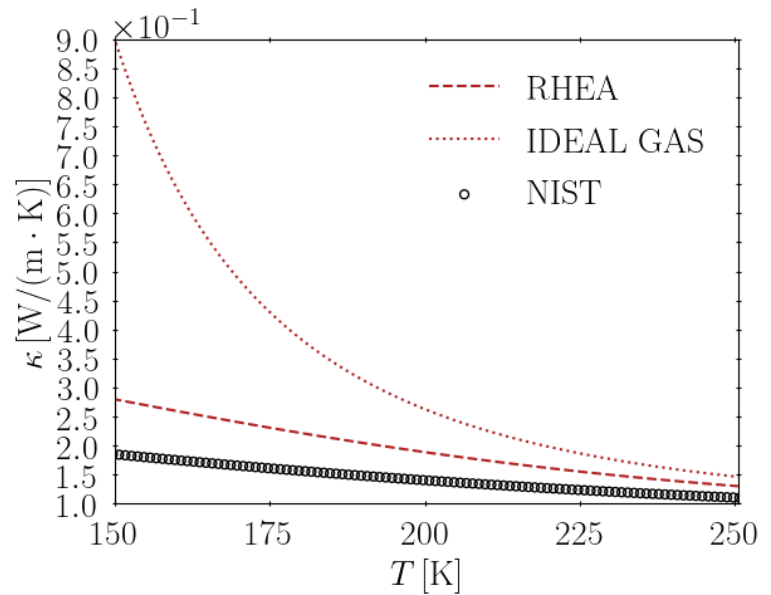
Internal Energy vs Temperature



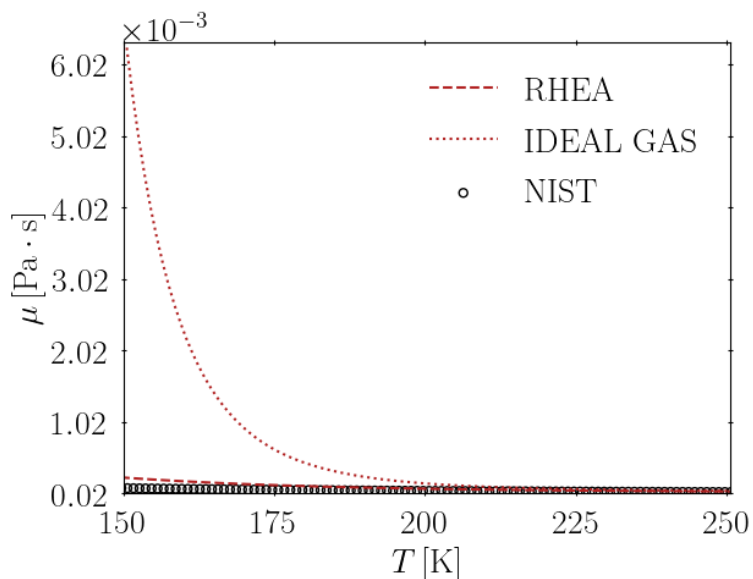
Sound speed vs Temperature



Thermal conductivity vs temperature

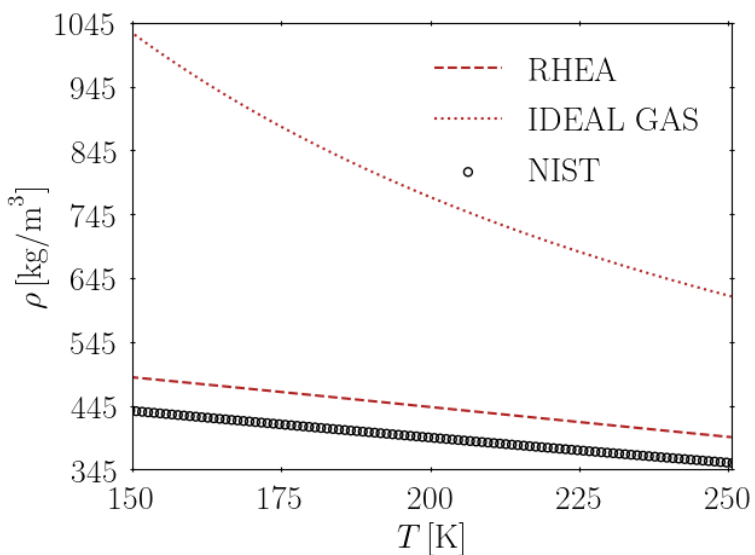


Viscosity vs Temperature

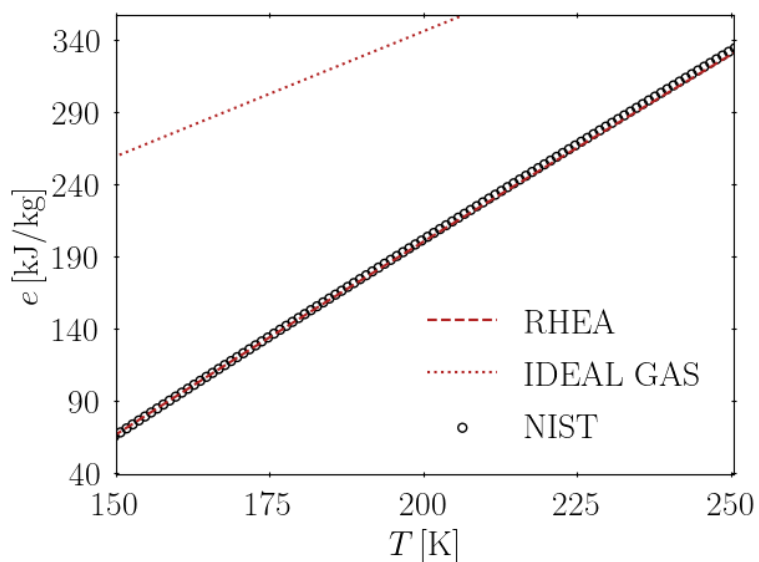


Metà (80 MPa)

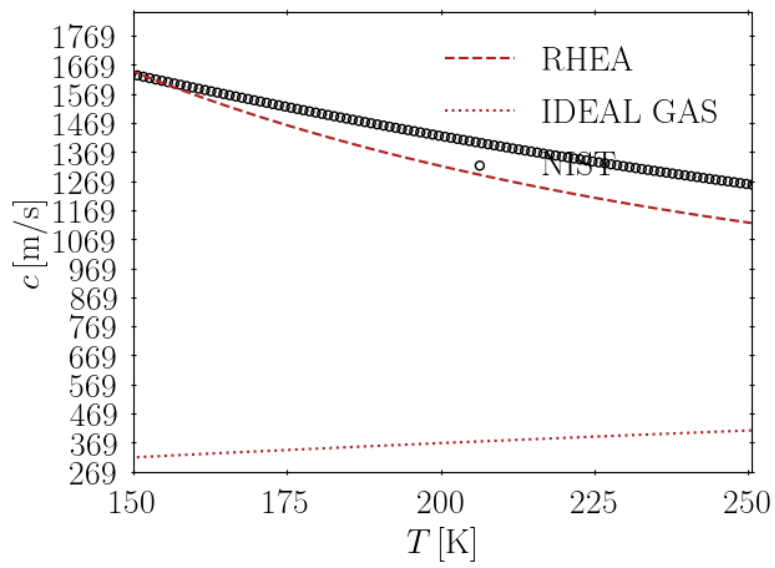
Density vs Temperature



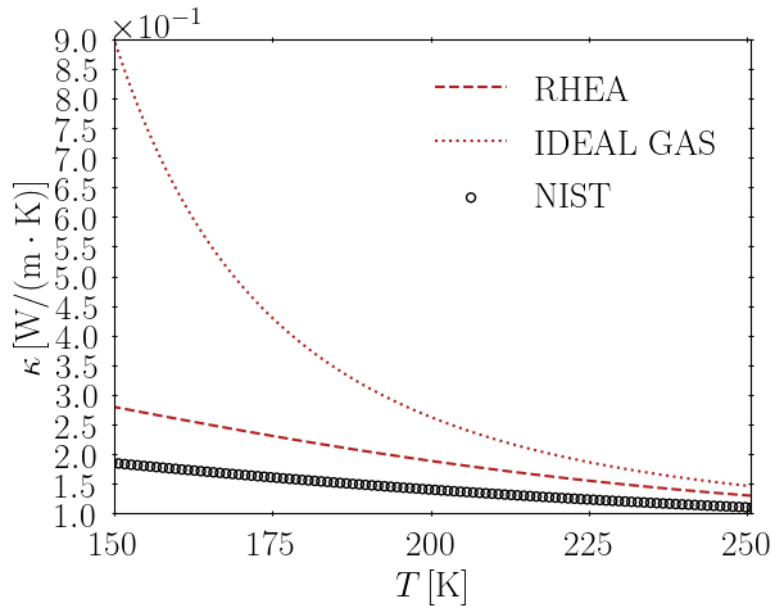
Internal Energy vs Temperature



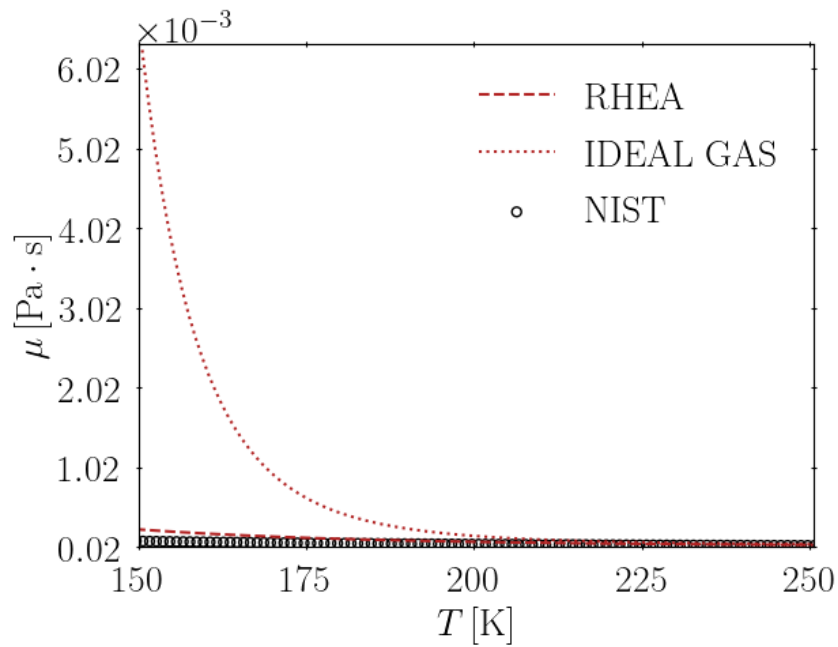
Sound speed vs Temperature



Thermal conductivity vs temperature

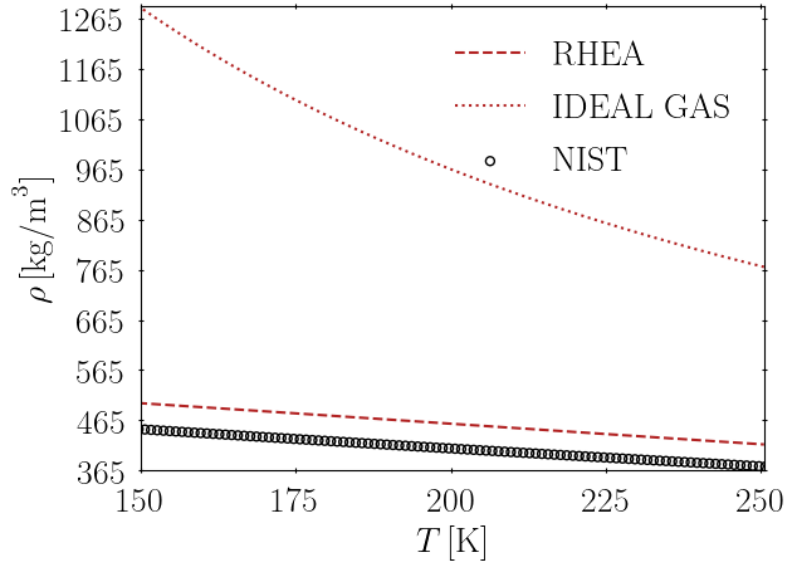


Viscosity vs Temperature

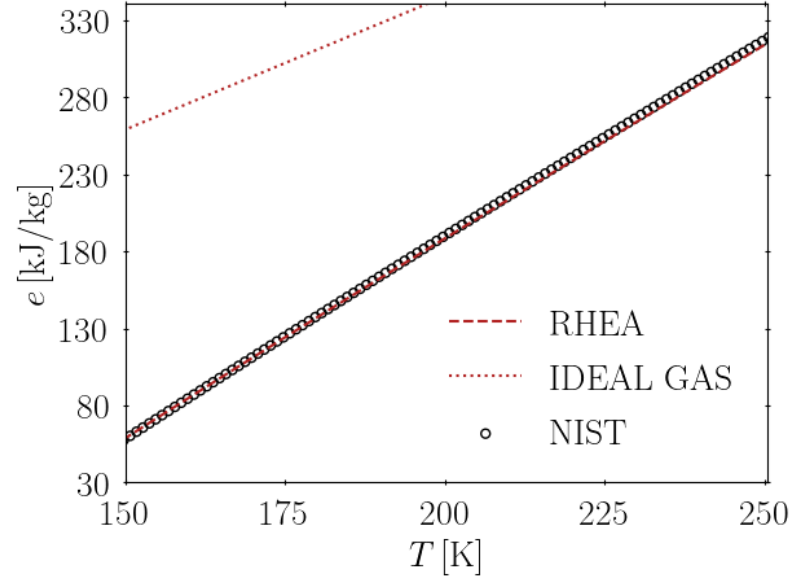


Metà (100 MPa)

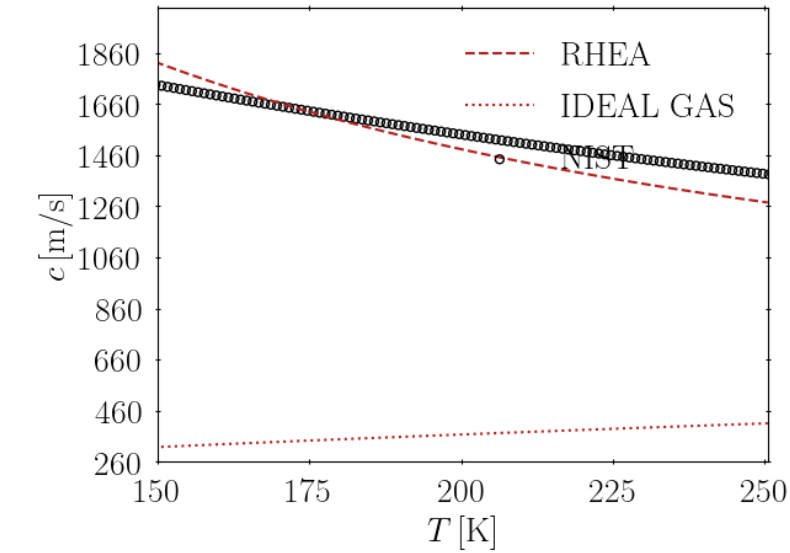
Density vs Temperature



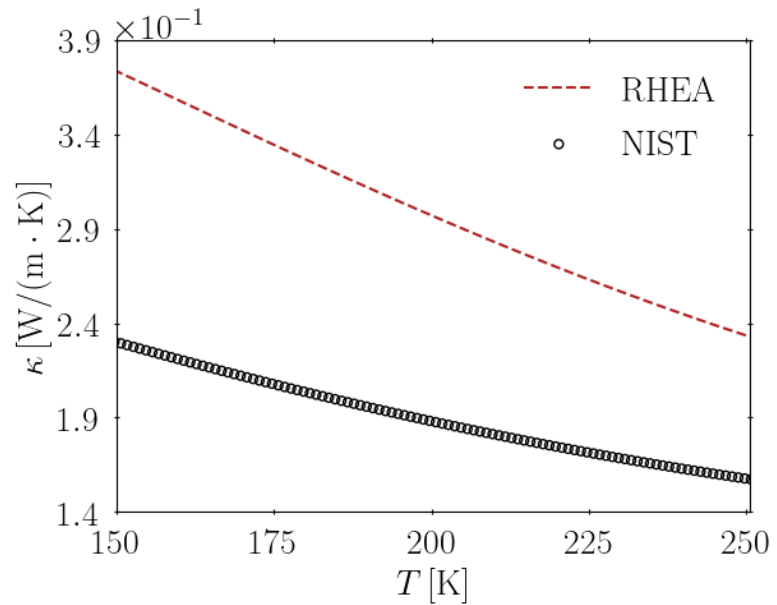
Internal Energy vs Temperature



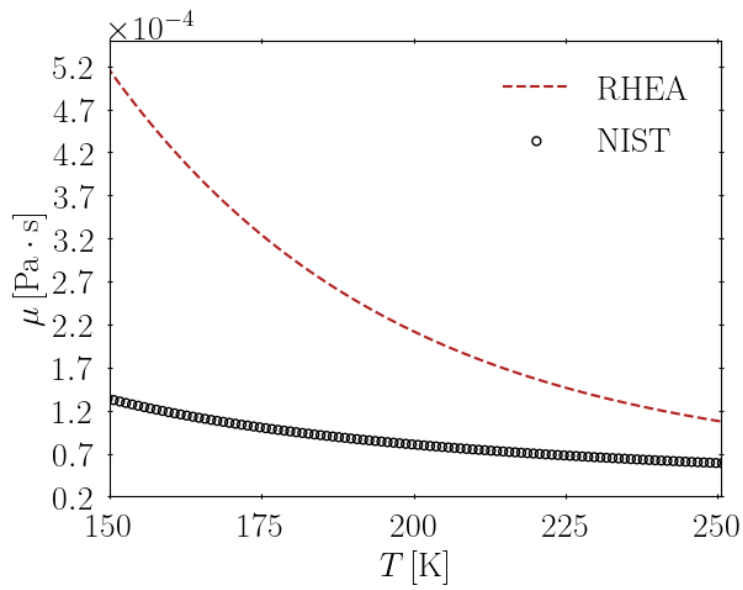
Sound speed vs Temperature



Thermal conductivity vs temperature

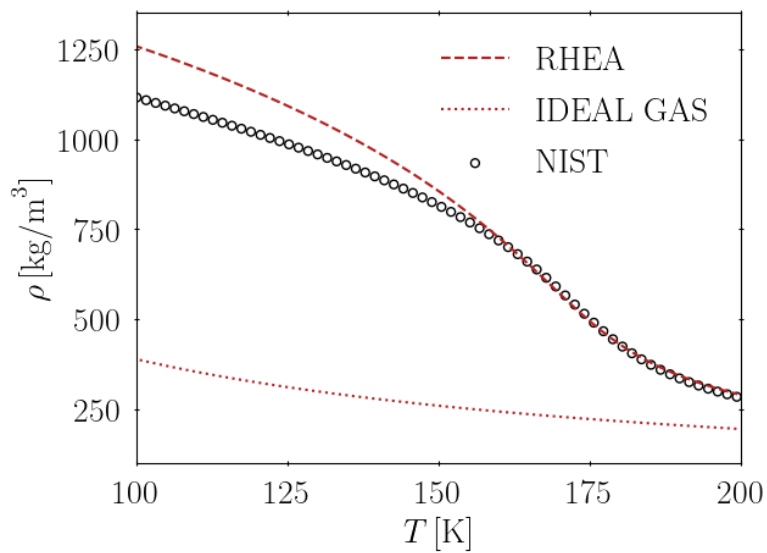


Viscosity vs Temperature

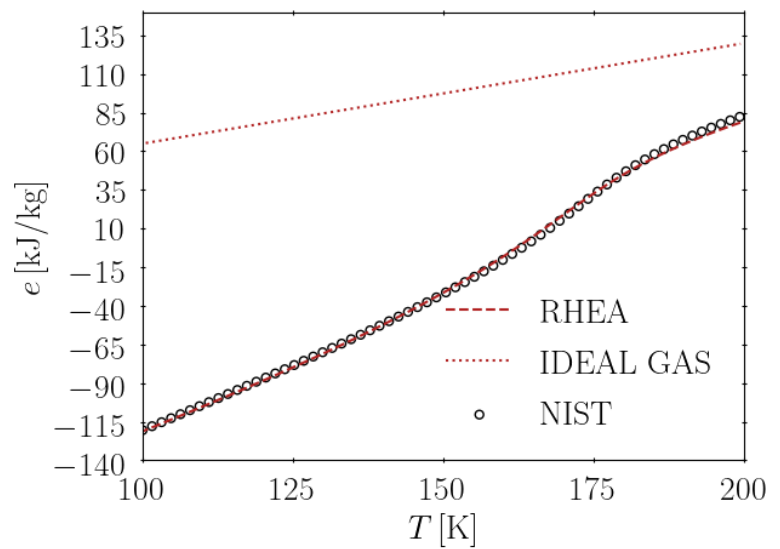


Oxigen (10,09 MPa)

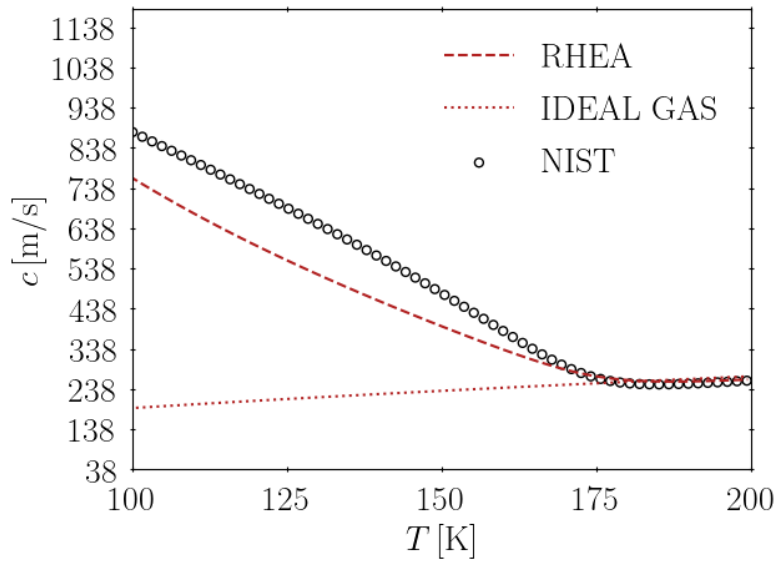
Density vs Temperature



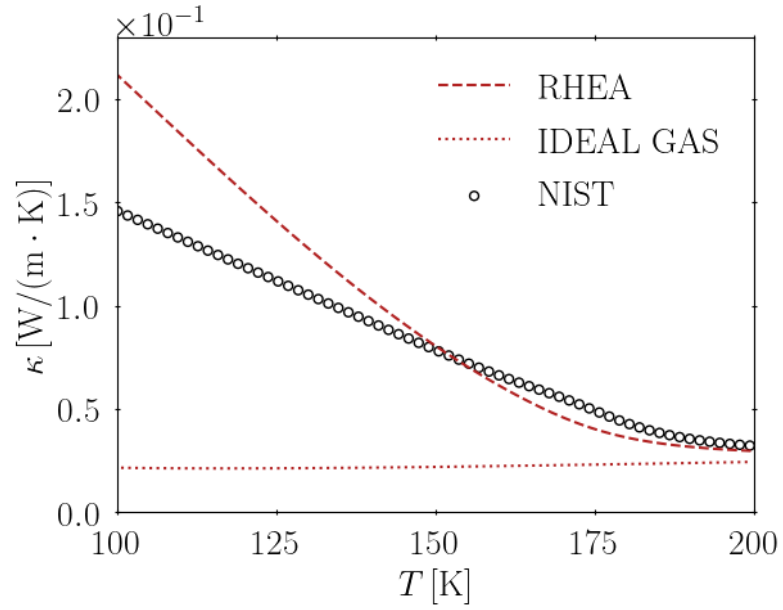
Internal Energy vs Temperature



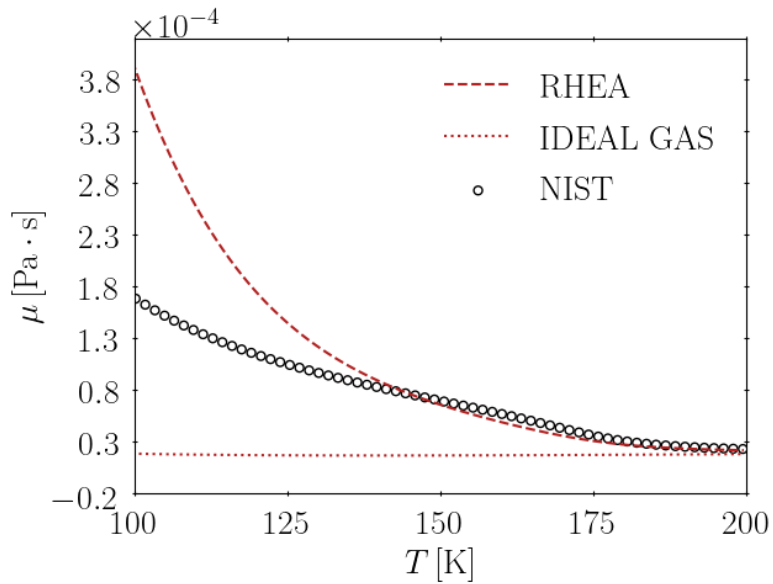
Sound speed vs Temperature



Thermal conductivity vs temperature

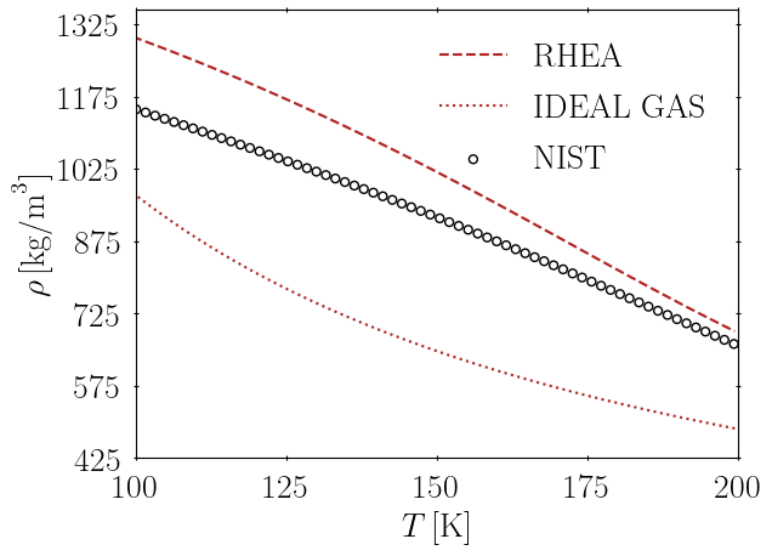


Viscosity vs Temperature

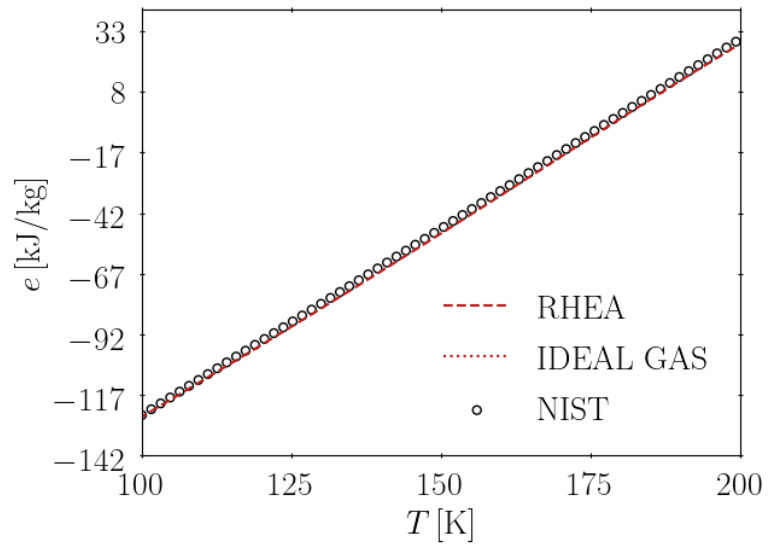


Oxigen (25,22 MPa)

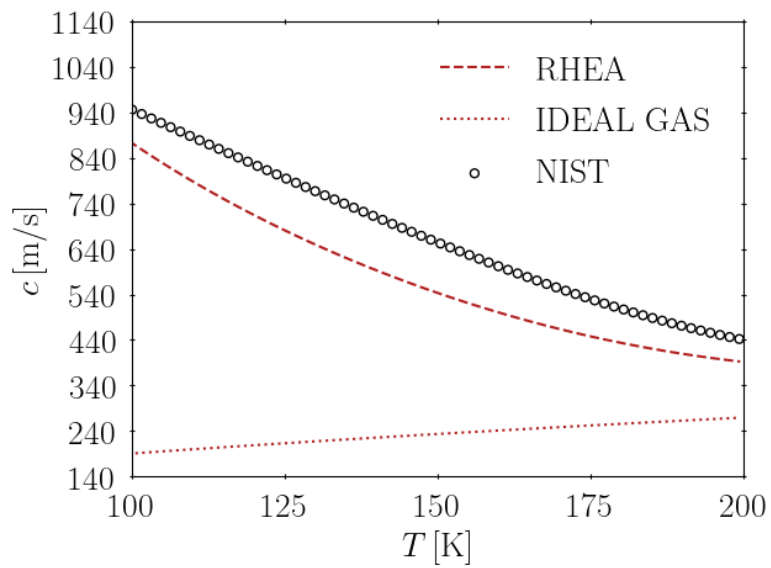
Density vs Temperature



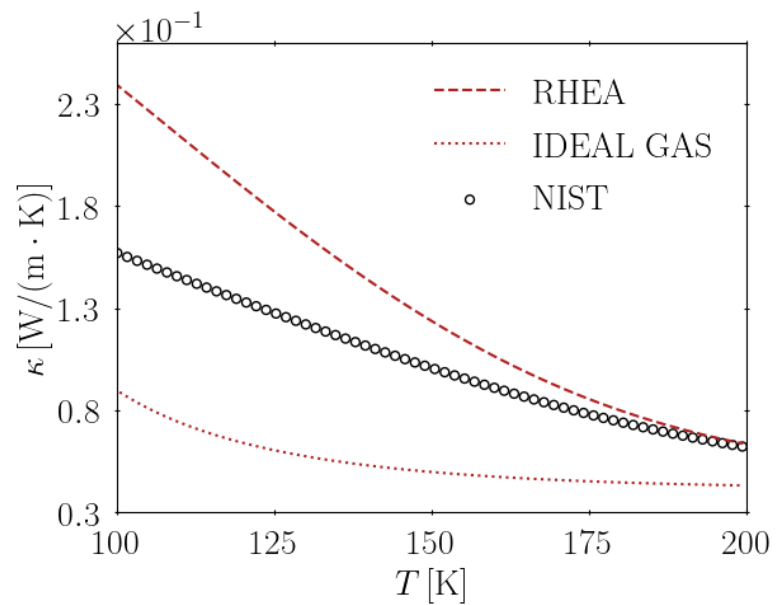
Internal Energy vs Temperature



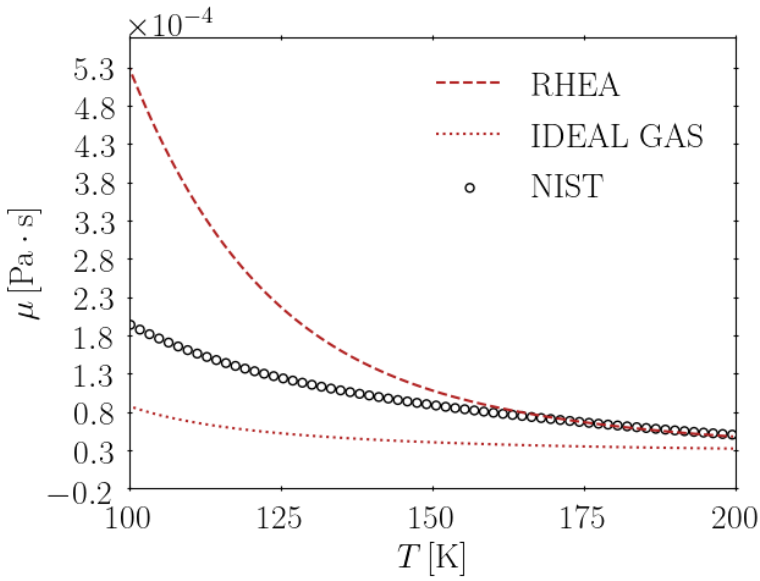
Sound speed vs Temperature



Thermal conductivity vs temperature

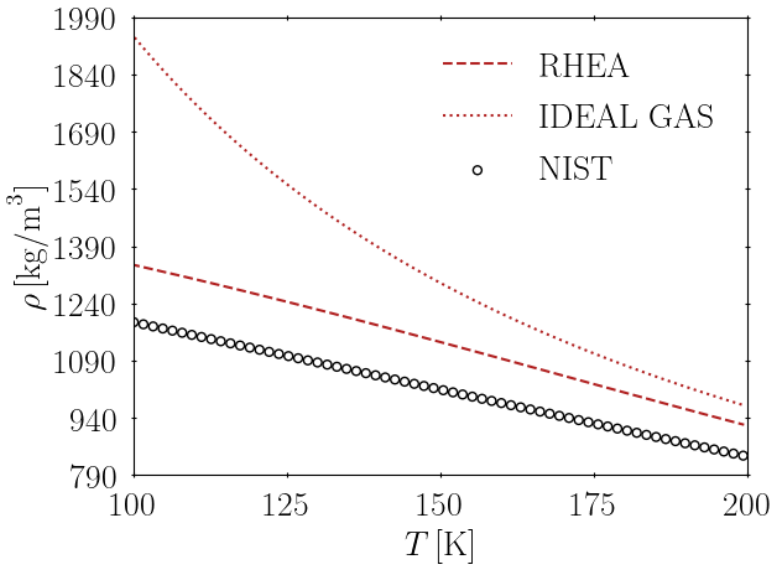


Viscosity vs Temperature

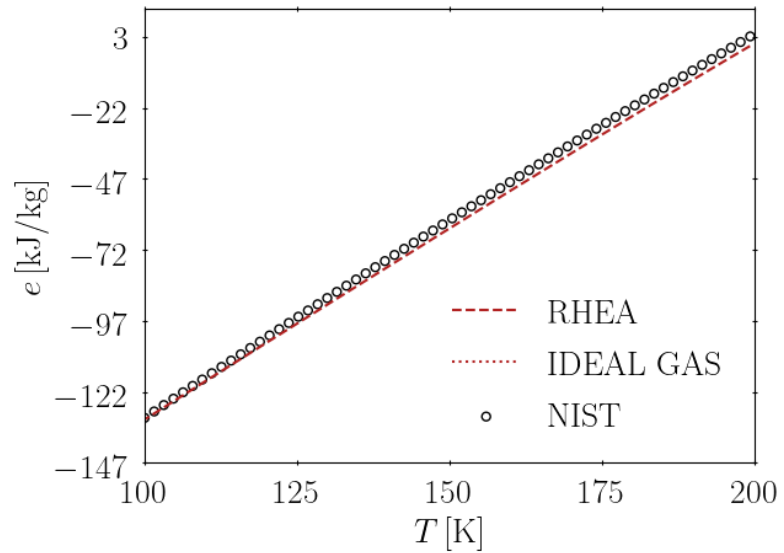


Oxygen (50,43 MPa)

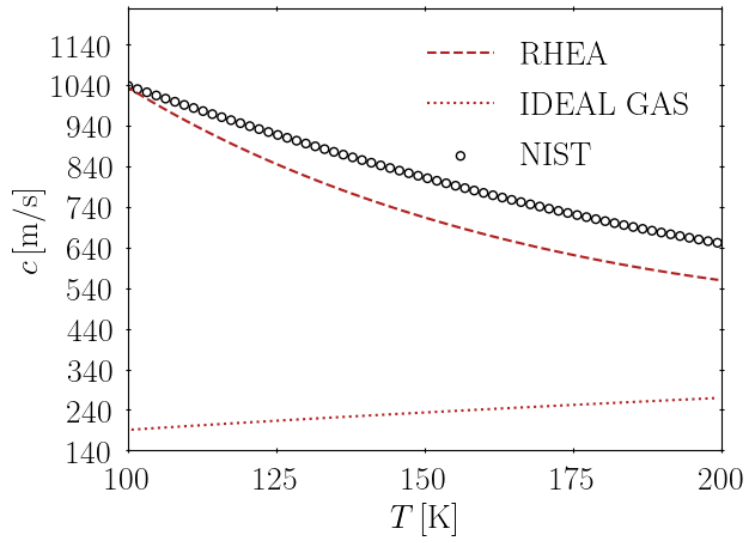
Density vs Temperature



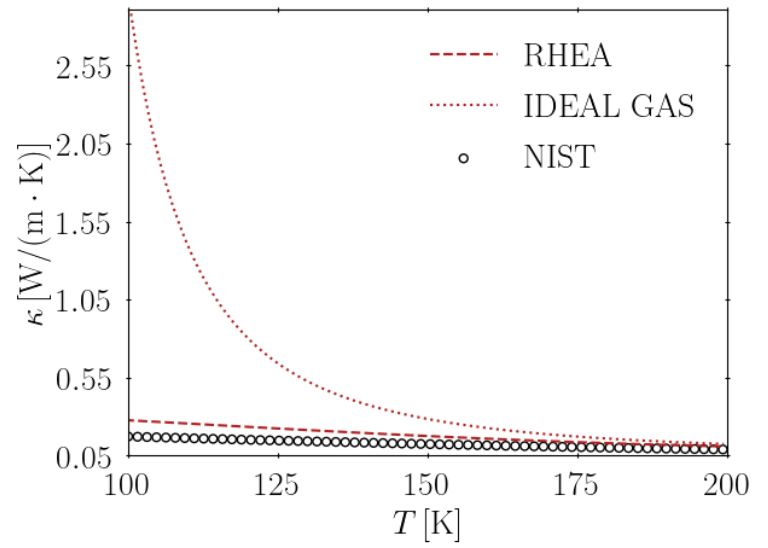
Internal Energy vs Temperature



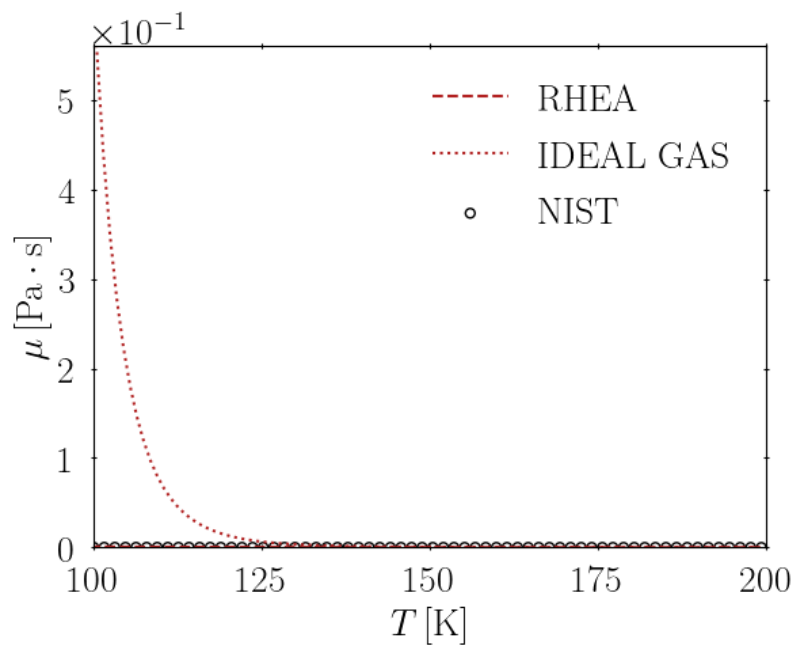
Sound speed vs Temperature



Thermal conductivity vs temperature

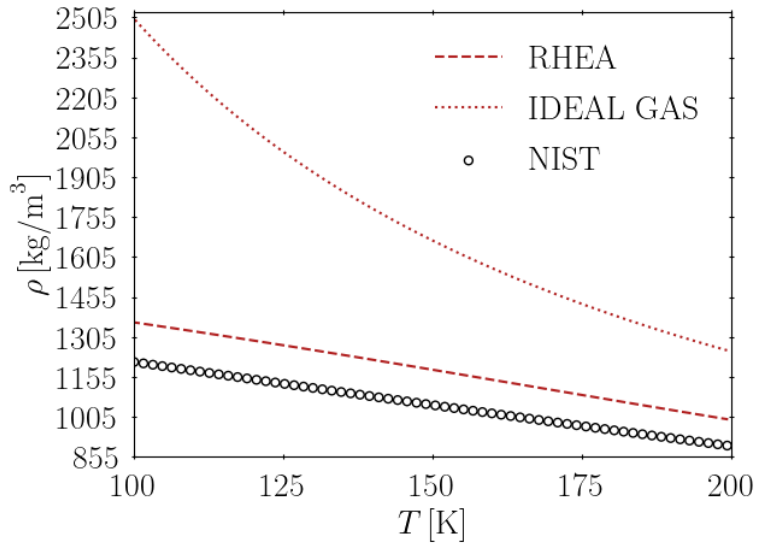


Viscosity vs Temperature

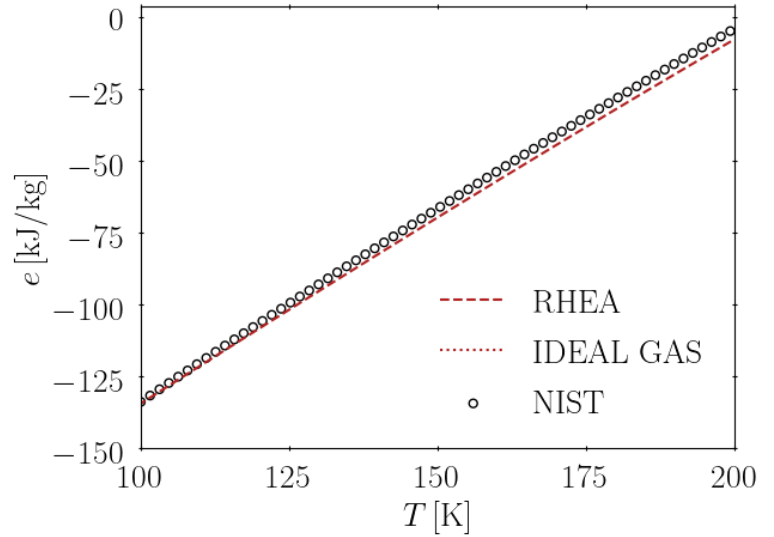


Oxigen (65 MPa)

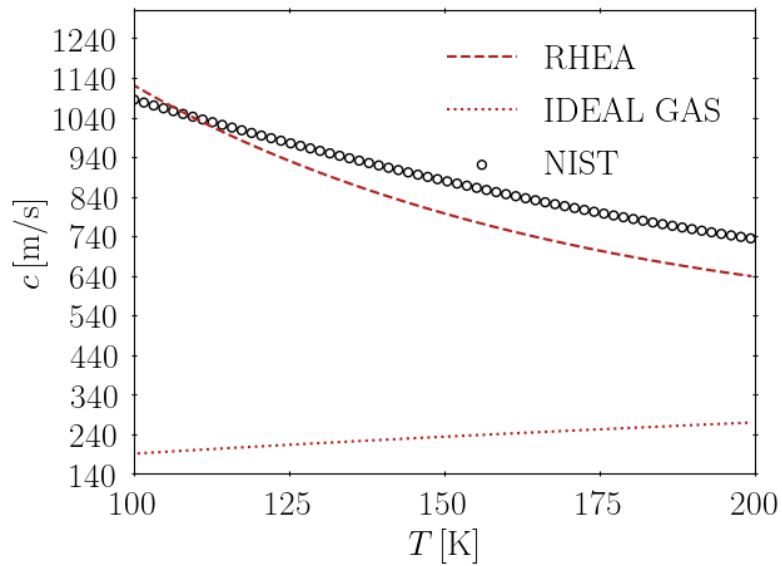
Density vs Temperature



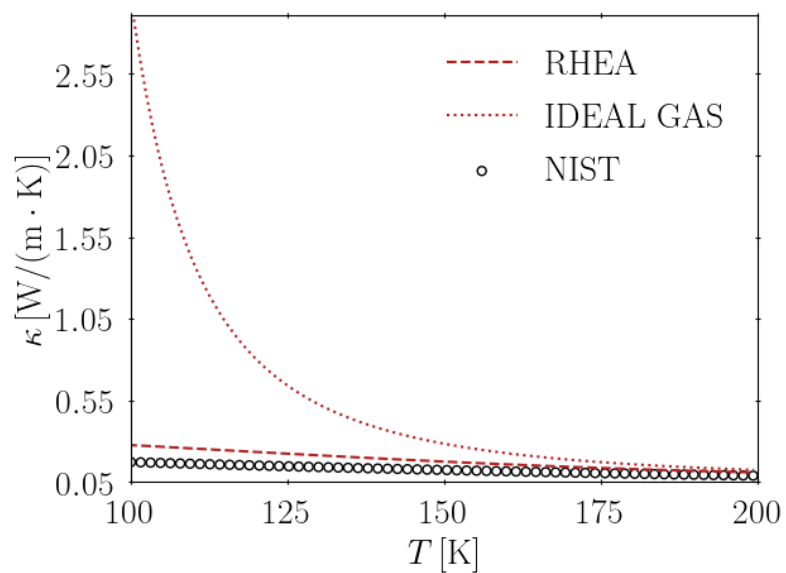
Internal Energy vs Temperature



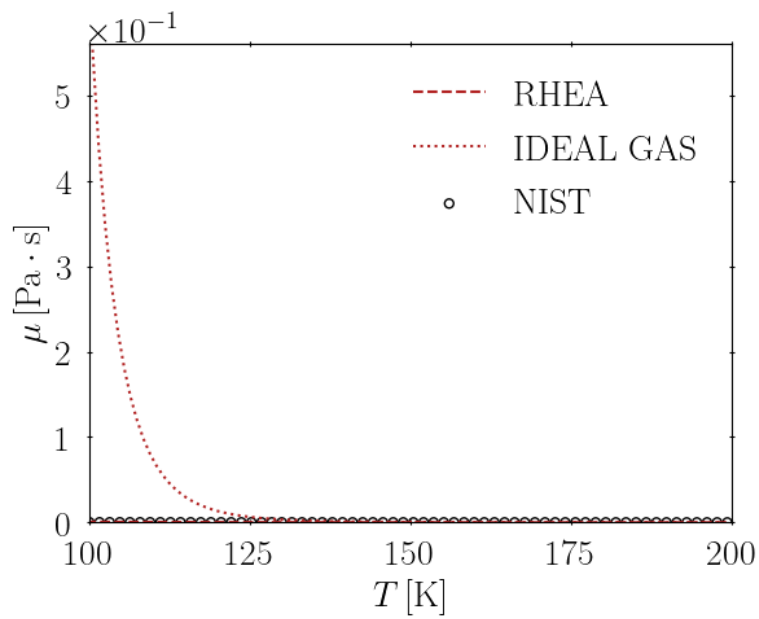
Sound speed vs Temperature



Thermal conductivity vs temperature

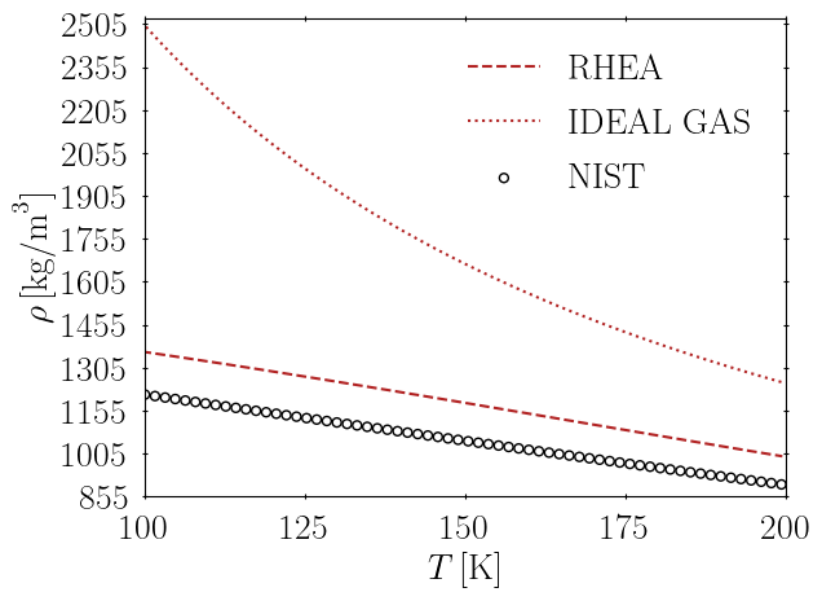


Viscosity vs Temperature

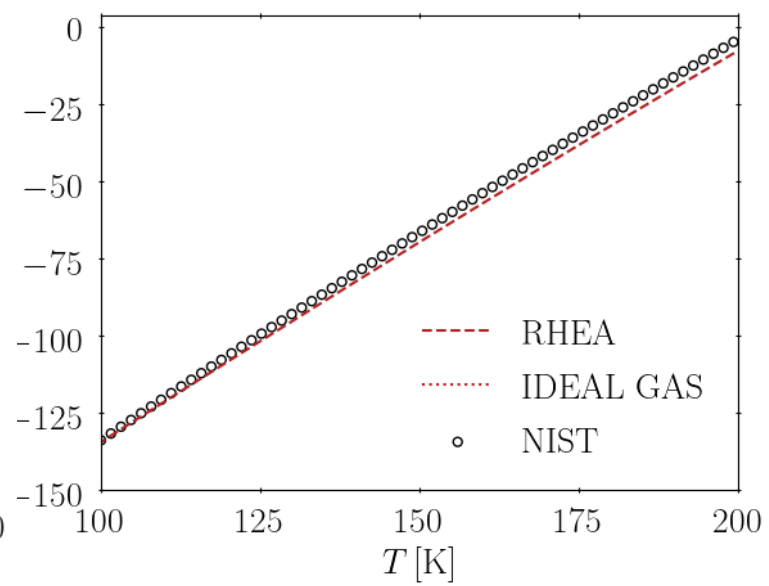


Oxygen (82 MPa)

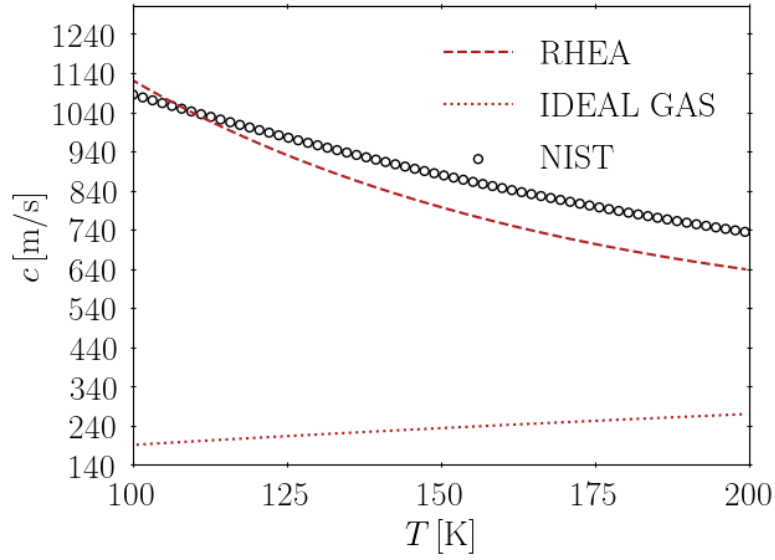
Density vs Temperature



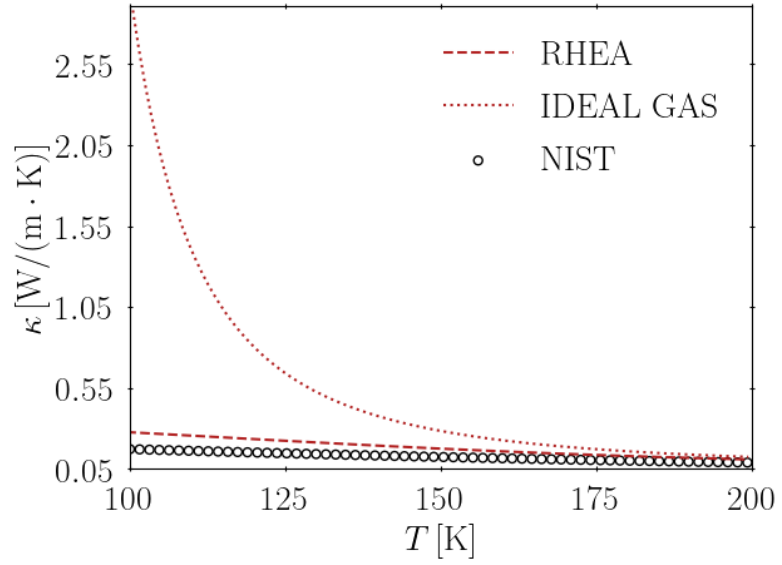
Internal Energy vs Temperature



Sound speed vs Temperature



Thermal conductivity vs temperature



Viscosity vs Temperature

