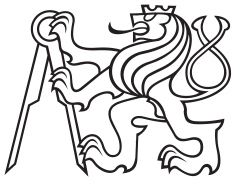


České vysoké učení technické v Praze
Fakulta Elektrotechnická

Program pro řešení dějů ve vodní páře

Diplomová práce

Autor: **Bc. Jindřich Bareš**

Vedoucí práce: **Ing. Petr Kočárník, Ph.D.**

Akademický rok: 2021/2022

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Bareš** Jméno: **Jindřich** Osobní číslo: **461493**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra elektrotechnologie**

Studijní program: **Elektrotechnika, energetika a management**

Specializace: **Technologické systémy**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Program pro řešení dějů ve vodní páře

Název diplomové práce anglicky:

Program for solving processes in water steam

Pokyny pro vypracování:

1. Proveďte rešerši problematiky stavové plochy vodní páry a formulace standardu IAPWS-97
2. V prostředí Matlab s využitím knihovny XSteam vytvořte skripty pro tisk t-s, h-s a dalších diagramů vodní páry
3. S využitím knihovny XSteam vytvořte v prostředí Matlab program pro řešení termodynamických dějů ve vodní páře

Seznam doporučené literatury:

Zaplatílek K., Doňar B.: Matlab tvorba uživatelských aplikací, BEN, 2004
Moran M., Shapiro H.: Fundamentals of Engineering Thermodynamics, Wiley, 2008
Cengel Y., Boles M.: Thermodynamics An Engineering Approach, McGraw Hill, 2019

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Petr Kočárník, Ph.D. katedra elektrických pohonů a trakce FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **03.02.2022** Termín odevzdání diplomové práce: _____

Platnost zadání diplomové práce: **30.09.2023**

Ing. Petr Kočárník, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

. | Datum převzetí zadání

Podpis studenta

Poděkování:
Děkuji vedoucímu práce Ing. Petru Kočárníkovi, Ph.D. za jeho četné a přínosné konzultace. Také děkuji svým blízkým za jejich trpělivost a podporu.

Čestné prohlášení:
Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 16. května 2022 Bc. Jindřich Bareš

Název práce: **Program pro řešení dějů ve vodní páře**

Autor: Bc. Jindřich Bareš

Studijní program: Elektrotechnika, energetika a management

Specializace: Technologické systémy

Druh práce: Diplomová práce

Vedoucí práce: Ing. Petr Kočárník, Ph.D.

Abstrakt: Hlavním cílem této práce bylo vytvořit program pro řešení dějů ve vodní páře. Pro tento účel byla provedena rešerše standardu IAPWS-IF97. Program byl vytvořen v prostředí Matlab za pomoci knihovny XSteam. Program řeší výpočty termodynamických veličin ve počátečním a koncovém stavu děje. Program také děje vykresluje do h-s, t-s a p-v diagramů. V rámci tohoto projektu byly vytvořeny i h-s a t-s diagramy vodní páry. Ty byly vytvořené pro tisk na formát papíru A3. Jejich škálované zmenšeniny jsou v příloze této práce.

Klíčová slova: Matlab, XSteam, IAPWS IF97, program, voda, pára

Title: **Program for solving processes in water steam**

*Author:*Bc. Jindřich Bareš

Abstract: Main goal of this thesis was to make a program for solving processes in water steam. For this purpose IAPWS-IF97 standard was researched. Program was written in Matlab environment using XSteam library. Program solves calculations of the thermodynamic variables in the initial and final state of the process. Program also plots the processes in h-s, t-s and p-v diagrams. Within this project h-s and t-s diagrams of water and steam were created. They were created to be printed on A3 paper. Scaled miniatures are in the apendix of this thesis.

Key words: Matlab, XSteam, IAPWS IF97, program, water, steam

Obsah

1. Seznam použitých veličin

Veličina	Jednotka	Veličina slovně
q	$(J \cdot kg^{-1})$	Měrné sdělené teplo
w_o	$(J \cdot kg^{-1})$	Měrná objemová práce
w_t	$(J \cdot kg^{-1})$	Měrná tlaková práce
t	$(°C)$	Teplota dle Celsiovy stupnice
T	(K)	Teplota absolutní
p	(bar)	Tlak
v	$(m^3 \cdot kg^{-1})$	Měrný objem
h	$(J \cdot kg^{-1})$	Měrná entalpie
s	$(J \cdot kg^{-1} \cdot K^{-1})$	Měrná entropie
x	$(-)$	Suchost
u	$(J \cdot kg^{-1})$	Měrná vnitřní energie

2. Úvod

Pro zobrazení dějů ve vodní páře jsou v termodynamice běžně používané diagramy h-s a t-s. Pro snadnější práci s těmito diagramy je vhodné vytvořit grafické rozhraní, které umí tyto děje vykreslit. Další možné využití příslušných skriptů lze najít při tisku těchto diagramů.

Hlavní motivací k vytvoření programu je neexistence volně dostupného softwaru, který umí tyto děje počítat podle standardu IAPWS-IF97. K dispozici je volně dostupný program Pára (Interaktivní grafický software pro termodynamické výpočty par) vytvořený na VUT Brno v devadesátých letech minulého století. Program byl vytvořen pro operační systém DOS, z čehož vplývá, že z dnešního pohledu má zastaralé uživatelské rozhraní a navíc termodynamické děje počítá dle staršího standardu IFC-67 (The 1967 IFC Formulation for Industrial Use) [1], [2].

3. Termodynamické děje a stavová plocha vodní páry

3.1 I. a II. hlavní věta termodynamiky

První věta termodynamiky je bilanční vyjádření zákona zachování energie. Může být napsána ve tvaru [3]

$$dq = du + dw + de_m, \tag{3.1}$$

kde dq je sdělené teplo, dw je mechanická práce vykonaná systémem (se znaménky dle znaménkové konvence), du představuje změnu vnitřní energie a de_m je energie nesená hmotou ve vstupních a výstupních kanálech. Pokud je uvažován uzavřený systém, pak energie de_m nesená hmotou je nulová a v takovém případě je možné první hlavní větu termodynamiky zapsat ve tvaru:

$$dq = du + dw, \tag{3.2}$$

Druhá věta termodynamiky říká, že tepelný stroj musí pracovat mezi dvěma tepelnými zásobníky s rozdílnými teplotami. Není možné, aby tepelný stroj pracoval pouze se zdrojem tepla, aniž by z něj teplo bylo odváděno [3]. Druhou hlavní větu termodynamiky je možné vyjádřit pomocí vztahu:

$$dq \leq T\, ds, \tag{3.3}$$

kde dq je sdělené teplo, T teplotu a ds změnu entropie.

3.2 Veličiny

Termodynamický systém je ohraničená množina objektu v pevném, kapalném i plynném skupenství a také někdy též ve stavu fázových přeměn. Vlastnosti termodynamického systému jsou vyjádřeny pomocí stavových veličin, pomocí kterých popisujeme tepelné děje které v látce probíhají. Ve vytvořeném programu pracujeme s následujícími stavovými veličinami: teplota, tlak, měrný objem, měrná entalpie, měrná entropie, suchost, měrná vnitřní energie soustavy. V programu pracujeme s veličinami v intenzitním tvaru tj. s veličinami vztaženými na $1kg$ pracovní látky (měrnými veličinami):

1. teplota $t\,(°C)$, $T\,(K)$,
2. tlak $p\,(Pa)$,
3. měrný objem $v\,(m^3 \cdot kg^{-1})$,
4. měrná vnitřní energie $u\,(J \cdot kg^{-1})$,
5. měrná entalpie $h\,(J \cdot kg^{-1})$,
6. měrná entropie $s\,(J \cdot kg^{-1} \cdot K^{-1}))$,
7. muchost $x\,[-]$.

3.3 Stav systému

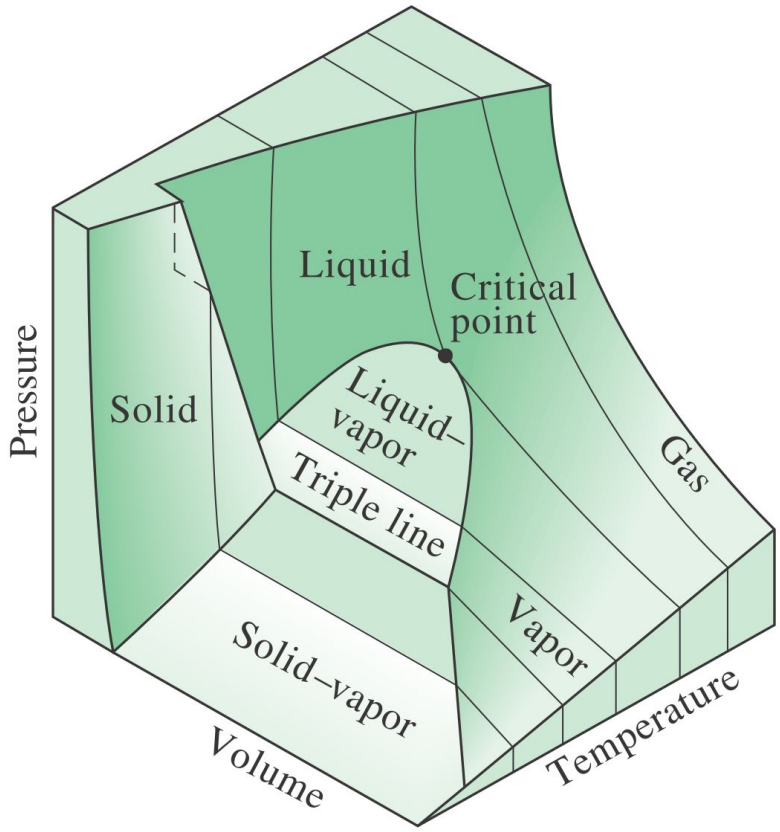
Stav systému je pak popsán stavovými veličinami. Počet stupňů volnosti termodynamického systému je dán Gibbsovým pravidlem:

$$n = k - f + 2, \tag{3.4}$$

kde pak n je počet stupňů volnosti systému, k je počet složek systému a f je počet fází systému. Pára je jednosložkový vícefázový systém. V oblasti mokré páry má 1 stupeň volnosti, v oblasti mokré páry má 2 stupně volnosti. Souvislost mezi n+1 stavovými veličinami vyjadřuje stavová rovnice.

3.4 Stavová plocha

Stavová plocha je grafickým vyjádřením stavové rovnice.



Obrázek 3.1: Termická stavová plocha vody $p = p(v, T)$. Převzato z [3]

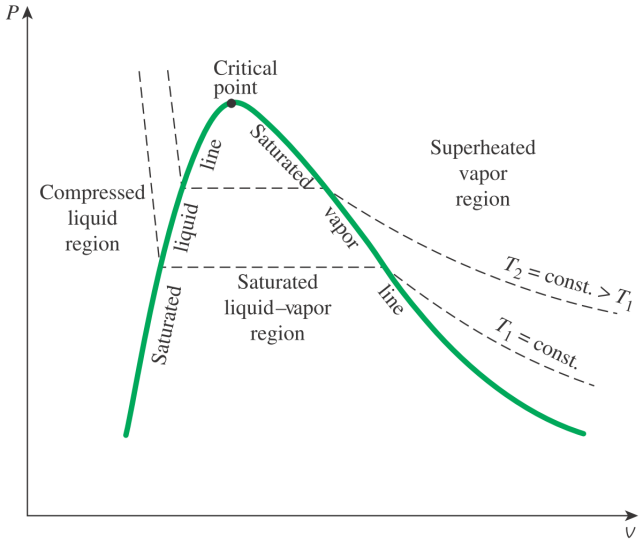
Počátek stavové plochy vody je umístěný do trojného bodu tr, jehož parametry jsou:

1. $T_{tr} = 273,16\ K$,
2. $p_{tr} = 611,657\ Pa$,
3. $\rho_{tr\_led} = 916,59\ kg \cdot m^3$,
4. $\rho_{tr\_voda} = 999,79\ kg \cdot m^3$,
5. $\rho_{tr\_para} = 0,004855\ kg \cdot m^3$.

Důležitou součástí stavové plochy je mezní křivka, která ohraničuje oblast tzv. mokré páry, kde se voda vyskytuje v kapalném a plynném skupenství. Další důležitý bod je kritický bod, který leží na vrcholu mezní křivky, jeho parametry jsou [4]:

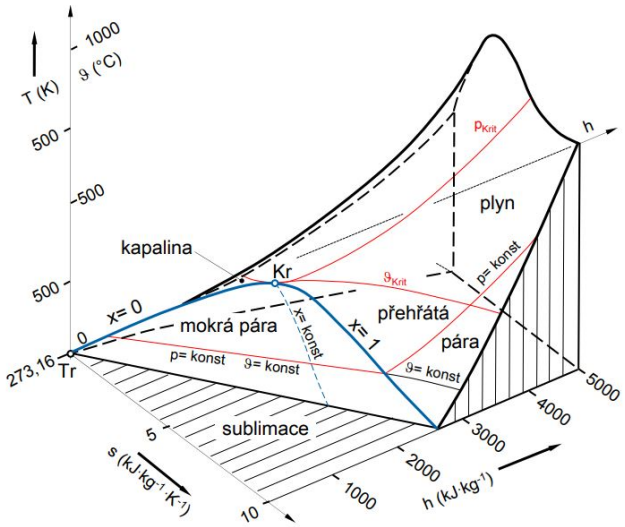
1. $T_{krit} = 674,096\ K$,
2. $p_{krit} = 22,064\ MPa$,
3. $\rho_{krit} = 322\ kg \cdot m^{-3}$.

Stavovou plochu je možné promítnout do tří souřadných rovin, tyto průměty označujeme jako diagramy. Příkladem může být průmět stavové plochy do roviny p-v, tj. p-v diagram:



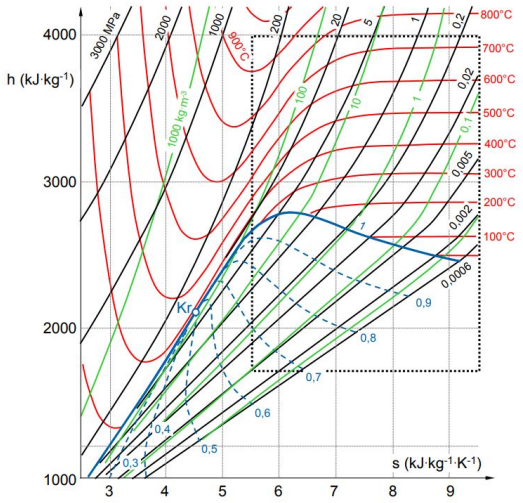
Obrázek 3.2: Znázornění p-v diagramu. Převzato z [3]

V těchto diagramech lze zakreslovat průběhy ostatních stavových veličin pomocí izočar, například čáry $t = konst.$. Stavovou plochu je možno také zakreslit pomocí kalorických veličin, kdy alespoň na jedné ose diagramu je vynesena kalorická veličina. Nejčastějším případem kalorické stavové plochy je průběh $T = T(h, s)$.

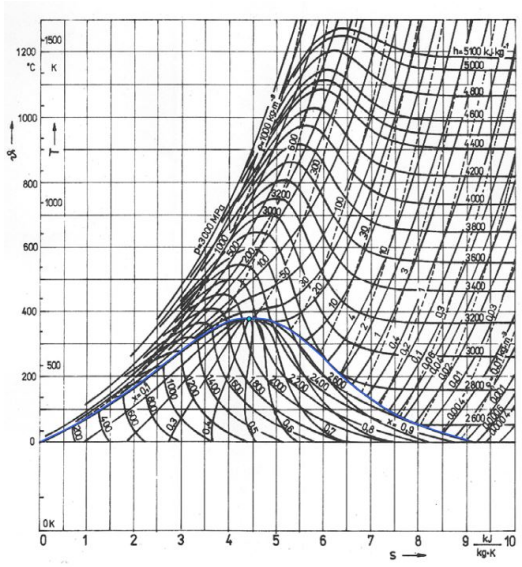


Obrázek 3.3: Kalorická stavová plocha vody $T = T(h, s)$. Převzato z [5]

Z průmětů této stavové plochy získáváme h-s a t-s diagram.



Obrázek 3.4: Stavová plocha vodí páry jako h-s diagram se znázorněním výřezu běžně používaného v energetické termodynamice. Převzato z [5]



Obrázek 3.5: Stavová plocha vodí páry jako t-s diagram. Převzato z [5]

3.5 Fázový přechod

Pro účely této práce je zajímavý stavový přechod kapalina plyn (pára). Proces fázového přechodu není okamžitý a je vysvětlen na příkladu níže.

Uvažujme kapalinu v nádobě uzavřené pístem. Pod pístem je pouze kapalina bez vzduchových bublin. Píst se může volně hýbat a působí na kapalinu tlakem 1 atmosféry. Uvažujme ohřev této kapaliny z pokojové teploty. Zprvu teplota poroste. Až do okamžiku kdy dosáhne 100 $°C$. Růst teploty je zastaven a začíná proces fázové přeměny. V tomto stavu je kapalina považována za sytou (saturovanou). Kapalina absorbuje energii (teplo) a mění se na plyn. Během fázové přeměny teplota neroste, neboť je veškerá dodaná energie spotřebována na fázový přechod. Látka se vyskytuje ve dvou skupenstvích a označujeme ji jako mokrou páru. Po ukončení fázového přechodu se látka nachází ve stavu syté (saturované) páry. Po odpaření veškeré kapaliny při dalším dodávání tepla teplota roste a stav látky označujeme jako přehřátá pára [3].

3.6 Děje ve vodní páře

Tato práce se nezabývá problematikou nerovnovážných termodynamických dějů. Pro sledování termodynamických změn v termodynamickém systému předpokládáme, že se tento systém nachází ve stavu lokální termodynamické rovnováhy. V takovém případě předpokládáme, že děj probíhá dostatečně pomalu a stavové veličiny jsou v celém termodynamickém systému vyrovnány. Tyto děje můžeme popsat i zobrazit v stavových diagramech. Pro účely termodynamiky většinou zkoumáme děje, kdy jedna ze stavových veličin zůstává konstantní. Jedná se o děj izotermický, izobarický, izochorický, izoentalpický a děj izoentropický. K nim je přiřazen děj adiabatický [4]:

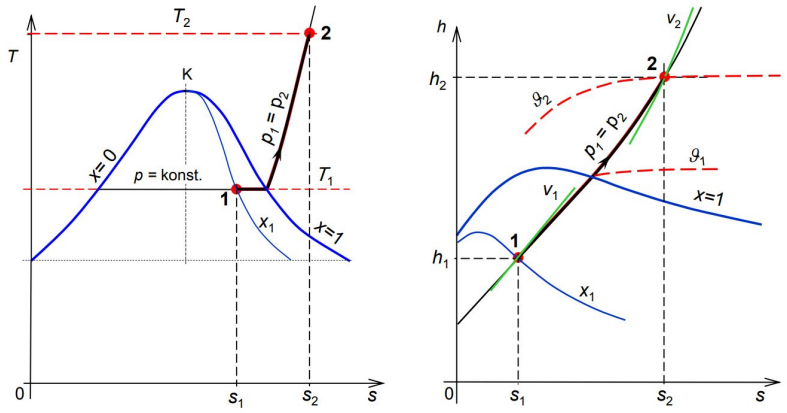
1. Izotermický děj je děj probíhající za konstantní teploty, $t = konstanta$,
2. Izobarický děj je děj probíhající za konstantního tlaku, $p = konstanta$,
3. Izochorický děj je děj probíhající za konstantního objemu, $v = konstanta$,

4. Izoentalpický děj je děj probíhající za konstantní entalpie, $h = konstanta$,

5. Izoentropický děj je děj probíhající za konstantní entropie, $s = konstanta$,

6. Adiabatický děj je děj, při kterém přes hranici soustavy není sděleno žádné teplo $q = 0$.

Pro znázornění práce s diagramem je zde uveden příklad izobarické stavové změny. Prvně je nutné mít definovaný počáteční stav. Například pomocí teploty t a suchosti x. Jedná se o dva nezávislé parametry a tedy je jimi stav jednoznačně definován. Pro znázornění izobarického děje je také nutné mít parametry koncového stavu. Respektive ze zadání izobarického děje pomocí počátečního stavu je známý tlak koncového stavu. Tedy pro jednoznačné určení koncového stavu děje stačí znát jeden další nezávislý parametr. Tím může být teplota. Znázornění izobarického děje v h-s a t-s diagramech pak vypadá následovně



Obrázek 3.6: Znázornění izobarického děje v t-s (vlevo) a h-s (vpravo) diagramech. Převzato z [5]

Konstrukce probíhá tak, že prvně je nalezen průsečík čáry suchosti reprezentující danou hodnotu a izotermy. Tento průsečík je počátečním stavem. Z diagramů je pak možné odečíst hodnoty ostatních termodynamických veličin. Pro znázornění děje a nalezení parametrů koncového stavu je pak z počátečního stavu vedena izobara. Její průsečík s izotermou koncového stavu pak představuje koncový stav. Opět je z tohoto průsečíku možné odečíst ostatní termodynamické veličiny.

Podobným postupem je možné znázornit i ostatní vratné děje.Pro adiabatický děj platí $q = 0$ a tedy $s = konstanta$. Odtud pak vychází i vykreslení tohoto děje do diagramů.

4. Matlab

Prostředí programu Matlab umožňuje dva způsoby zadávání příkazů. Prvním způsobem je zadávání pomocí příkazového řádku (konzole). Tento způsob je snadný a rychlý, ale umožňuje zadávání příkazů pouze ručně. Tím pádem je náročné pomocí příkazového řádku vytvořit komplikovanější script, funkci nebo celý program. Druhou velkou nevýhodou je, že tato metoda neumožňuje snadné uložení příkazu pro další opakované spuštění.

Druhým způsobem zadávání je editor příkazů. Je možné použít libovolný textový editor mimo prostředí Matlabu, ale obecně to není doporučeno. Vestavěný editor prostředí Matlab je vybaven řadou nástrojů, které usnadňují a zpříjemňují tvorbu sad příkazů. Příkladem takových nástrojů je zaprvé automatické ukládání souboru pro případ neočekávaného pádu vývojového prostředí, za druhé obsáhlá nápověda (help) a za třetí kvalitní nástroje pro ladění kódu, které umožňují velmi snadno a rychle nalézt syntaktickou chybu.

Po sepsání souboru příkazů v editoru prostředí Matlab je uživateli umožněno tento soubor uložit pod jím zvoleným názvem. Přípona souboru takto vytvořeného je .m, neboli takzvaný m-soubor. Tento soubor je možné dále upravovat a ladit. Jednotlivé m-soubory mohou být vzájemně spouštěny a tím mohou vytvářet složitější programové struktury [6].

4.1 m-soubory

m-soubory mohou být dvou typů, a to script nebo funkce. Script je v prostředí Matlabu seznam příkazů spustitelný z příkazové řádky. Pokud uvnitř scriptu není definováno jinak, tak nepotřebuje žádné další vstupní parametry. Script je možné pojmenovat libovolně.

Druhou možností je uložit seznam příkazů jako Matlabovskou funkci, která musí na prvním řádku začínat klíčovým slovem *function*. m-soubor může obsahovat těchto funkcí více, přičemž první funkce v pořadí je tzv. funkce hlavní, která je viditelná z ostatních m-souborů, ostatní funkce jsou funkce vedlejší, které jsou vidět pouze v rámci m-souboru, ve kterém jsou deklarované.

4.2 Anonymní funkce

Mimo zmíněný typ funkce v prostředí Matlab existují i další typy funkcí. Jedním z nich jsou anonymní funkce (anonymous functions). Její definice je v rámci kódu (skriptu), kde je taky používána. Definice funkce je na jednom řádku. Definice této funkce obecně vypadá následovně:

```
name = @ (arglist) expr
```

Zde *name* představuje název funkce, *arglist* seznam vstupních argumentů a *expr* matematický výraz funkce[7]. Triviálním příkladem takové definice je:

```
fun = @ (x) x^2
```

Tu je pak možné volat pomocí:

```
vysledek = fun(2)
```

Toto volání do proměnné *vysledek* přiřadí hodnotu $2^2 = 4$.

4.3 Funkce funkce

Existuje mnoho případů, kdy je nutné, aby funkce nepracovala s proměnnou, ale s celou funkcí. Příkladem takového případu je funkce Matlab-u *fzero*, která nalezne hodnotu x libovolné funkce tak, aby se hodnota funkce v tomto bodě rovnala nule. Neboli $f(x) = 0$. Vstupem takovéto funkce musí být jiná funkce. To se v prostředí Matlab nazývá funkce funkce.

K předání funkce do funkce funkce je možné využít odkazy na funkce (function handle). Jedná se hodnotu (data type) prostředí Matlab, která je s funkcí provázaná a je možné ji využít pro volání této funkce. Tedy v případě předání tohoto odkazu do jiné funkce je možné tento odkaz využít pro volání funkce s odkazem provázané.

Tento odkaz je možné vytvořit pomocí symbolu @. Například $@fun$ vytvoří odkaz na funkci *fun*. Tento odkaz je pak možné přiřadit k proměnné a předávat dál:

```
funHandle = @fun
```

Pro anonymní funkce není nutné tvořit tyto odkazy neboť jejich definice samotná již tento odkaz vytváří.

Pro účely využití vstupní funkce v rámci funkce funkce je v její definici funkce vyjádřena pomocí náhradního jména (dummy name). Pod proměnnou tohoto náhradního jména funkce je uložen odkaz na funkci. Tu je pak v rámci funkce funkce možné volat pomocí této proměnné.

Například funkce, která vykreslí funkci je funkce funkce. Pomocí dříve uvedeného příkladu *fun* (anonymní funkce) je pak možné objekt funkce funkce *vykresliFunkci* definovat následovně:

```
function out = vykresliFunkci(FIn)
% vykresliFunkci vykreslí FIn v intervalu [-1,1]

x = linspace(-1,1);
y = arrayfun(FIn,x);
out = plot(x,y);
end
```

Další možností volání funkce v rámci funkce funkce je volání pomocí jména funkce. V takovém případě je při volání funkce funkce do příkazu vložen název funkce jako string (text). Funkce je pak v rámci funkce funkce volána pomocí příkazu Matlab-u *feval* [7]. Například:

```
function out = vykresliFunkci(FIn)
% vykresliFunkci vykreslí FIn v intervalu [-1,1]

x = linspace(-1,1);
y = feval(FIn,x);
out = plot(x,y);
end
```

Tato funkce by pak pro vykreslení funkce sinus byla volána:

```
figura = vykresliFunkci('sin')
```

4.4 Vedlejší funkce

Uživatelem definované funkce mohou být v prostředí Matlab velmi složité. Pro jejich zjednodušení mohou být užitečné takzvané vedlejší funkce (sub functions). Pod funkce jsou funkce definované v rámci m souboru mimo hlavní funkci, nebo skript tohoto souboru. Obvykle se jedná o jednoduché, často opakované operace. Triviálním příkladem souboru funkce s definovanou pod funkcí je:

```
function out = vykresliFunkci(FIn)
% vykresliFunkci vykreslí FIn v intervalu [-1,1]
[x, y] = vypoctiBody(-1,1)
out = plot(x,y);
end

function [x,y] = vypoctiBody(a,b)
x = linspace(a,b);
y = feval(FIn,x);
end
```

V tomto případě je hlavní funkcí m-souboru funkce *vykresliFunkci*. Ta pak volá svou pod funkci *vypoctiBody*, která je definovaná v tomtéž souboru pod hlavní funkcí. Pod funkce je možné volat pouze v rámci souboru, ve kterém jsou definovány [7].

4.5 Vnořené funkce

Vnořené funkce (nested functions) jsou funkce definované ve funkci. Pomocí vnořených funkcí je možné definovat funkci ve funkci. Tyto definice mohou mít mnoho vrstev a tím mohou být velmi matoucí. Proto je zde pouze naznačen triviální příklad vnoření jedné funkce do funkce druhé.

```
function y = A(a1,a2)
...
    function z = B(b1,b2)
    ...
    end
...
end
```

Vnoření funkce do funkce má určité specifické vlastnosti. Těmi jsou sdílení proměnných (workspace) a možnost volání funkce vnořené z funkce hlavní a naopak [7].

4.6 Grafické prvky

V rámci prostředí Matlab je možné vytvářet uživatelské rozhraní. Základním prvkem, takovým "obalem", všech grafických prvků je figura (figure). V rámci figury je pak možné definovat další prvky jako jsou tlačítka, výběrové seznamy, plochy grafů a jiné. Jinak řečeno figura je okno, které uživatel vidí. Ostatní prvky jsou pak tlačítka, texty a další prvky uvnitř tohoto okna. Každá figura má svou sadu vlastností, které je možné měnit a nastavovat. Těchto vlastností je mnoho například barva figury, rozměry nebo poloha, ve které se na obrazovce počítače otevře. Všechny prvky přiřazené k figuře mají své sady vlastností.

Významnou položkou prvků figury (např. tlačítek) je takzvaný callback, což je odkaz na funkci, která se vykoná např. při stisku tlačítka myši, pokud je její ukazatel uvnitř podřízených objektů [6].

4.7 App designer

Důležitým nástrojem pro tvorbu uživatelských aplikací je interaktivní vývojové prostředí App Designer. Toto prostředí poskytuje plně integrovanou verzi Matlab editoru a velkou škálu interaktivních komponent uživatelského rozhraní. Prostředí také nabízí rozložení komponent pomocí mřížky a automatické škálování aplikace při změně zobrazovacího zařízení (monitoru) [8].

Prostředí umožňuje dva pohledy na aplikaci. Jeden je pohled grafický. Ten je velmi užitečný pro návrh geometrie rozmístnění prvků, volbu jejich barvy či velikosti. Není nutné nastavovat jednotlivá tlačítka tak, aby byla stejně daleko od okraje. To za nás zařídí mřížka. Velmi podobné je to i u ostatních grafických záležitostí prvků aplikace.

Druhý pohled na použití App desginer-u spočívá v pohledu na aplikaci v kódu. Prostředí App Designéru nám neumožňuje automaticky generovaný kód ručně změnit. Kód je na šedém pozadí a nelze ho přepsat. Tento přístup je užitečný pro definice jednotlivých funkcí. Například pokud chceme vytvořit aplikaci o jednom tlačítku a jednom grafu. Chceme aby, na stisk tlačítka byla v osách vykreslen průběh funkce $y = x$. Pak si pomocí App Designéru a jeho grafického pohledu na aplikaci snadno určíme vzájemnou polohu prvků tlačítka a plochy grafu, určíme text na tlačítku, velikost textu a jeho font. V tomto okamžiku ale tlačítko nic nedělá a plocha grafu je prázdná.

Nyní máme dvě možnosti. První z nich je si v prostředí Matlab-u vytvořit m-soubor s funkcí tlačítka. To však nemusí být výhodné, protože to přináší komplikace při přenosu proměnných a vlastností ostatních prvků. Druhou možností je v prostředí App Designer-u vytvořit nový callback na akci, v našem případě stisk tlačítka. Zde nám prostředí App Designer-u umožní funkci callback stisknutí tlačítka vyplnit naším kódem.

Po vyplnění funkce callback-u v prostředí soubor uložíme. Přípona souboru vytvořeného pomocí prostředí App Designer je *.mlapp*.

5. Standard IAPWS IF97

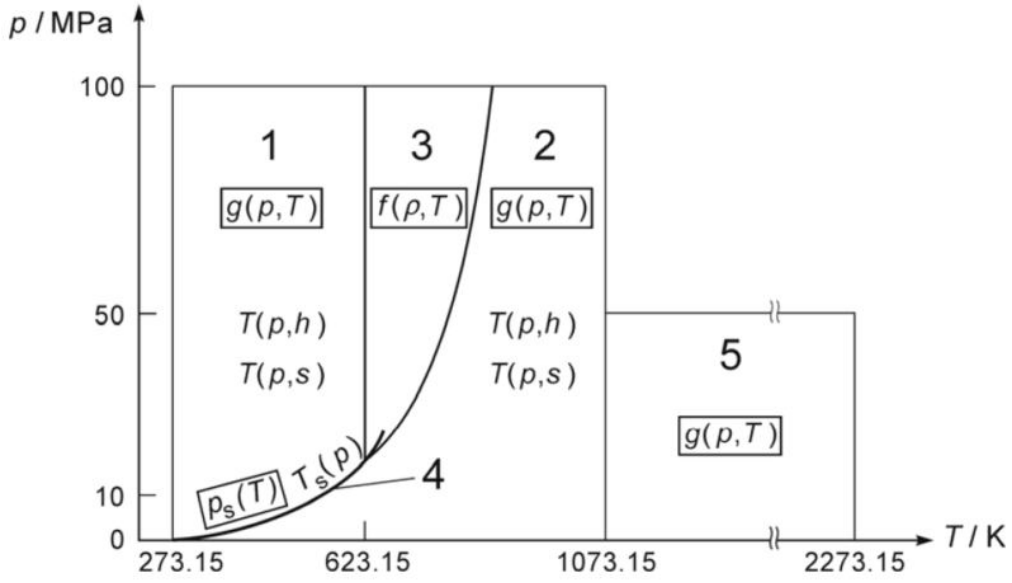
Standard je dostupný na stránkách organizace The International Association for the Properties of Water and Steam (dále jen IAPWS). Členy této mezinárodní organizace jsou Argentina, Brazílie, Británie, Irsko, Kanada, Česká Republika, Dánsko, Francie, Německo, Řecko, Itálie, Japonsko, Rusko, USA a spolupracuje se Švýcarskem [9].

Odkaz ke stažení dokumentu standardu IAPWS Industrial Formulation 1997 (dále jen IAPWS IF97) je dostupný z [10]. Jedná se o dokument o 49 stránkách popisující formulace v jednotlivých oblastech (regionech). Dokument byl revidován v letech 2007, 2009, 2010 a 2012. Poslední revize formulaci IAPWS IF97 rozšířila oblast 5 [10].

Cílem této práce není dokument přeložit. Proto je v této práci pouze uveden úvod k dokumentu a pro ilustraci problematiky definice jedné oblasti.

5.1 IAPWS IF97 - úvod

Definice standardu IAPWS IF97 je rozdělená do pěti oblastí. Oblasti jsou zobrazené na obr.5.1. Každá z oblastí je definována jinými rovnicemi. Pro teploty mezi $273,15\ K$ ($0\ °C$) a $1073,15\ K$ ($800\ °C$) je definovaný tlak $0\ MPa$ až $100\ MPa$. A pro teploty $1073,15\ K$ ($800\ °C$) až $2273,15\ K$ ($2000\ °C$) je pak definovaný tlak nižší, a to až $50\ MPa$ [10]. Oblasti i s jejich mezemi platnosti jsou k graficky znázorněné na obr.5.1.

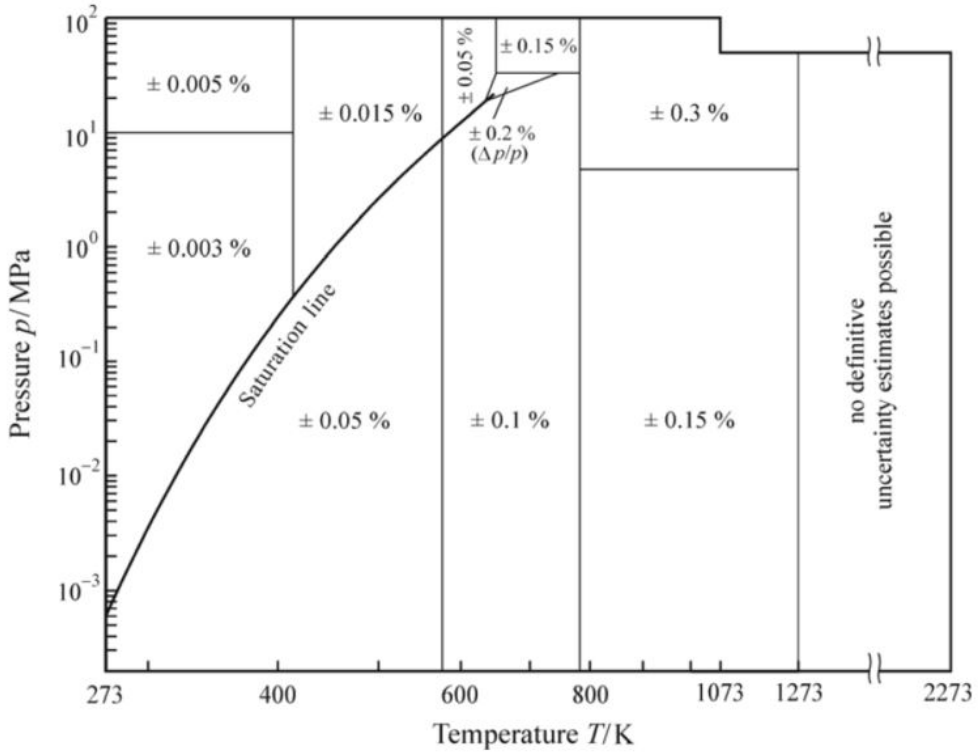


Obrázek 5.1: IAPS IF97 oblasti.

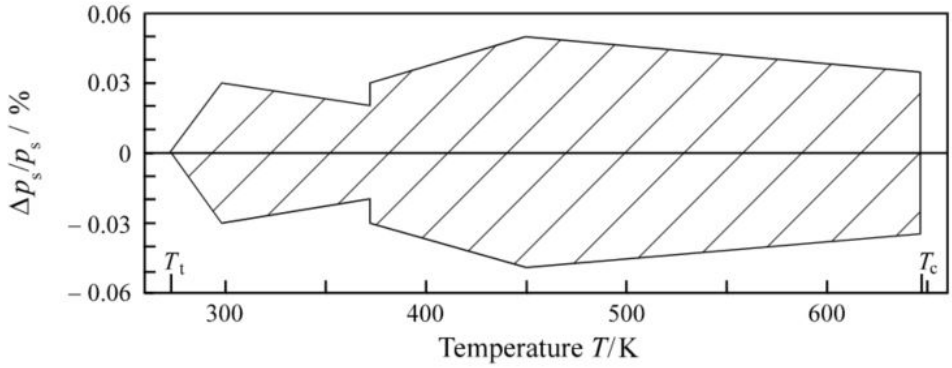
kde p je tlak, T je absolutní teplota, h je entalpie, s je entropie a ρ je hustota vody, respektive vodní páry. Definice oblastí 1, 2 a 5 $g(p,T)$ je Gibbsova volná energie, oblasti 3 $f(\rho,T)$ je Hemholtzova volná energie a oblasti 4 $p_s(T)$ je saturační-tlaková rovnice [10].

Dále jsou definovány takzvané zpětné rovnice $T(p,h)$, $T(p,s)$ a $T_s(p)$. Tyto rovnice jsou v souladu s rovnicemi z předchozí definice standardu.Výhodou zpětných rovnice je že umožňují rychlé výpočty parametrů vody, respektive vodní páry [10]. Pro tuto práci je důležité, že se jedná o funkce v závislosti na tlaku a entropii a funkce v závislosti na tlaku a entalpii.

Standard IAPWS IF97 obsahuje také odhady nejistot. Nejistoty byly porovnávány s předchozím standardem IAPWS-95 (IAPWS Formulation 1995 for the Thermodynamic Properties of Ordinary Water Substance for General and Scientific Use). Dokument uvádí grafy zobrazující nejistoty objemu, isobarické tepelné kapacity, rychlosti zvuku a saturačního tlaku [10]. Pro účely této práce jsou relevantní pouze saturační tlak a objem.



Obrázek 5.2: Graficky znázorněné nejistoty výpočtu objemu. Porovnání IAPS IF97 a IAPWS-95. Chybu v oblasti 5 nelze určit. Dřívější definice standardu neměla tuto oblast definovanou [10].



Obrázek 5.3: Graficky znázorněné nejistoty výpočtu saturačního tlaku. Porovnání IAPS IF97 a IAPWS-95 [10].

5.2 Oblast 1

5.2.1 Rovnice

Základní rovnicí pro oblast 1 je Gibbsova volná energie g. Rovnice je uvedena v bezrozměrné formě $\gamma = g/(RT)$, kde γ je Gibbsova volná energie, T je absolutní teplota a R je měrná plynová konstanta. Rovnice pak vypadá následovně:

$$\frac{g(p,T)}{RT} = \gamma(\pi,\tau) = \sum_{i=1}^{34}(7,1-\pi)^{I_i} n_i (\tau - 1,22)^{J_i}, \tag{5.1}$$

kde $\pi = p/p^*$, p^* je $16,53\ MPa$, $\tau = T^*/T$, T^* je $1386\ K$ a $R = 0,461526\ kJ \cdot kg^{-1} \cdot K^{-1}$. Dále pak n_i, I_i a J_i vycházejí z tabulky 5.1. Všechny termodynamické veličiny vody a vodní páry mohou být pomocí rovnice 5.1 a jejích derivací odvozeny. Vztahy mezi termodynamickými veličinami, γ a příslušných derivací jsou shrnuty v tabulkách 5.2 a 5.3 [10].

Od páté mezinárodní konference na téma vlastnosti vodní páry v Londýně 1956 je vnitřní energie a entropie vody v trojném bodě nastavena na hodnotu nula [10]:

$$u'_t = 0\ ;\ s'_t = 0 \tag{5.2}$$

Aby byla tato podmínka splněna, tak musejí být teplota a tlak v trojném bodě následující [10]:

$$T_t = 273,16\ K\ ;\ p_t = 0611,657\ Pa \tag{5.3}$$

Tím z rovnice 5.1 získáme entalpii saturované kapaliny v trojném bodě [10]:

$$h'_t = 0,611783\ J \cdot kg^{-1} \tag{5.4}$$

i	I_i	J_i	n_i	i	I_i	J_i	n_i
1	0	–2	0.146 329 712 131 67	18	2	3	–0.441 418 453 308 46 $\times 10^{-5}$
2	0	–1	– 0.845 481 871 691 14	19	2	17	–0.726 949 962 975 94 $\times 10^{-15}$
3	0	0	– 0.375 636 036 720 40 $\times 10^{1}$	20	3	–4	–0.316 796 448 450 54 $\times 10^{-4}$
4	0	1	0.338 551 691 683 85 $\times 10^{1}$	21	3	0	–0.282 707 979 853 12 $\times 10^{-5}$
5	0	2	– 0.957 919 633 878 72	22	3	6	–0.852 051 281 201 03 $\times 10^{-}9$
6	0	3	0.157 720 385 132 28	23	4	–5	–0.224 252 819 080 00 $\times 10^{-}5$
7	0	4	– 0.166 164 171 995 01 $\times 10^{-1}$	24	4	–2	–0.651 712 228 956 01 $\times 10^{-6}$
8	0	5	0.812 146 299 835 68 $\times 10^{-3}$	25	4	10	–0.143 417 299 379 24 $\times 10^{-12}$
9	1	–9	0.283 190 801 238 04 $\times 10^{-3}$	26	5	–8	–0.405 169 968 601 17 $\times 10^{-6}$
10	1	–7	– 0.607 063 015 658 74 $\times 10^{-3}$	27	8	–11	–0.127 343 017 416 41 $\times 10^{-8}$
11	1	–1	– 0.189 900 682 184 19 $\times 10^{-1}$	28	8	–6	–0.174 248 712 306 34 $\times 10^{-9}$
12	1	0	– 0.325 297 487 705 05 $\times 10^{-1}$	29	21	–29	–0.687 621 312 955 31 $\times 10^{-18}$
13	1	1	– 0.218 417 171 754 14 $\times 10^{-1}$	30	23	–31	0.144 783 078 285 21 $\times 10^{-19}$
14	1	3	– 0.528 383 579 699 30 $\times 10^{-4}$	31	29	–38	0.263 357 816 627 95 $\times 10^{-22}$
15	2	–3	– 0.471 843 210 732 67 $\times 10^{-3}$	32	30	–39	–0.119 476 226 400 71 $\times 10^{-22}$
16	2	0	– 0.300 017 807 930 26 $\times 10^{-3}$	33	31	–40	0.182 280 945 814 04 $\times 10^{-23}$
17	2	1	0.476 613 939 069 87 $\times 10^{-4}$	34	32	–41	–0.935 370 872 924 58 $\times 10^{-25}$
12	1	0	– 0.325 297 487 705 05 $\times 10^{-1}$	29	21	–29	–0.687 621 312 955 31 $\times 10^{-18}$
13	1	1	– 0.218 417 171 754 14 $\times 10^{-1}$	30	23	–31	0.144 783 078 285 21 $\times 10^{-19}$
14	1	3	– 0.528 383 579 699 30 $\times 10^{-4}$	31	29	–38	0.263 357 816 627 95 $\times 10^{-22}$
15	2	–3	– 0.471 843 210 732 67 $\times 10^{-3}$	32	30	–39	–0.119 476 226 400 71 $\times 10^{-22}$
16	2	0	– 0.300 017 807 930 26 $\times 10^{-3}$	33	31	–40	0.182 280 945 814 04 $\times 10^{-23}$
17	2	1	0.476 613 939 069 87 $\times 10^{-4}$	34	32	–41	–0.935 370 872 924 58 $\times 10^{-25}$

Tabulka 5.1: Tabulka koeficientů bezrozměrné rovnice pro Gibbsovu energii [10].

Veličina	Vztah
$\nu = (\partial g/\partial p)_T$	$\nu(\pi,\tau)\frac{p}{RT} = \pi\gamma_\pi$
$u = g - T(\partial g/\partial T)_p - p(\partial g/\partial p)_T$	$\frac{u(\pi,\tau)}{RT} = \tau\gamma_\tau - \pi\gamma_\pi$
$s = -(\partial g/\partial T)_p$	$\frac{s(\pi,\tau)}{R} = \tau\gamma_\tau - \gamma$
$h = g - T(\partial g/\partial T)_p$	$\frac{h(\pi,\tau)}{RT} = \tau\gamma_\tau$
$c_p = (\partial h/\partial T)_p$	$\frac{h(\pi,\tau)}{R} = -\tau^2\gamma_{\tau\tau}$
$c_v = (\partial u/\partial T)_\nu$	$\frac{c_v(\pi,\tau)}{R} = -\tau^2\gamma_{\tau\tau} + \frac{(\gamma_\pi - \tau\gamma_{\pi\tau})^2}{\gamma_{\pi\pi}}$
$w = \nu(-\partial p/\partial\nu)_s)^{1/2}$	$\frac{w^2=(\pi,\tau)}{RT} = \frac{\gamma_\pi^2}{\frac{(\gamma_\pi - \tau\gamma_{\pi\tau})^2}{\tau^2\gamma_{\pi\pi}} - \gamma_{\pi\pi}}$

$$\gamma_\pi = [\frac{\partial\gamma}{\partial\pi}]_\tau \ , \ \gamma_{\pi\pi} = [\frac{\partial^2\gamma}{\partial\pi^2}]_\tau \ , \ \gamma_\tau = [\frac{\partial\gamma}{\partial\tau}]_\pi \ , \ \gamma_{\tau\tau} = [\frac{\partial^2\gamma}{\partial\tau^2}]_\pi \ , \ \gamma_{\pi\tau} = [\frac{\partial^2\gamma}{\partial\pi\partial\tau}]$$

Tabulka 5.2: Vztahy mezi termodynamickými veličinami a funkcí bezrozměrné Gibbsovy volné energie γ a jejími derivacemi [10].

$$\gamma = \sum_{i=1}^{34} n_i(7, 1-\pi)^{I_i}(\tau - 1,222)^{J_i}$$

$$\gamma_\pi = \sum_{i=1}^{34} -n_i I_i(7, 1-\pi)^{I_i - 1}(\tau - 1,222)^{J_i}$$

$$\gamma_{\pi\pi} = \sum_{i=1}^{34} n_i I_i(I_i - 1)(7, 1-\pi)^{I_i - 2})(\tau - 1.222)^{J_i}$$

$$\gamma_\tau = \sum_{i=1}^{34} n_i(7, 1-\pi)^{I_i} J_i(\tau - 1,222)^{J_i - 1}$$

$$\gamma_{\tau\tau} = \sum_{i=1}^{34} n_i(7, 1-\pi)^{I_i} J_i(J_i - 1)(\tau - 1,222)^{J_i - 2}$$

$$\gamma_{\pi\tau} = \sum_{i=1}^{34} -n_i I_i(7, 1-\pi)^{I_i - 1} J_i(\tau - 1,222)^{J_i - 1}$$

$$\gamma_\pi = [\frac{\partial\gamma}{\partial\pi}]_\tau \ , \ \gamma_{\pi\pi} = [\frac{\partial^2\gamma}{\partial\pi^2}]_\tau \ , \ \gamma_\tau = [\frac{\partial\gamma}{\partial\tau}]_\pi \ , \ \gamma_{\tau\tau} = [\frac{\partial^2\gamma}{\partial\tau^2}]_\pi \ , \ \gamma_{\pi\tau} = [\frac{\partial^2\gamma}{\partial\pi\partial\tau}]$$

Tabulka 5.3: Bezrozměrná Gibbsova volná energie γ a její derivace [10].

Rovnice jsou platné v oblasti 1 znázorněného na obrázku 5.1. Tj. od teploty $273,15\,K$ po $623,15\,K$ a od tlaku $p_s(T)$ po $100\,MPa$ [10].

5.2.2 Zpětné rovnice

Dle dokumentu IAPWS IF97 jsou zpětné rovnice dvě. Výpočty za pomocí obou dvou funkcí probíhají bez iterací. Jedná se o funkci tlaku a entalpie (p, h) a funkci tlaku a entropie (p, s) [10]. Zpětná rovnice $T(p, h)$ pro oblast je má následující bezrozměrnou formu

$$\frac{T(p,h)}{T^*} = \theta(\pi\eta) = \sum_{i=1}^{20} n_i \pi^{I_i} (\eta + 1)^{J_i} \tag{5.5}$$

kde $\theta = T/T^*$, $pi = p/p^*$ a $\eta = h/h^*$ s $T^* = 1\ K$, $p^* = 1\ MPa$ a $h^* = 2500 Kj\ kg^{-1}$ [10].

i	I_i	J_i	n_i	i	I_i	J_i	n_i
1	0	0	-0.238 724 899 245 21$\times 10^{3}$	11	1	4	-0.659 647 494 236 38 $\times 10^{1}$
2	0	1	0.404 211 886 379 45 $\times 10^{3}$	12	1	10	0.939 654 008 783 63 $\times 10^{-2}$
3	0	2	0.113 497 468 817 18 $\times 10^{3}$	13	1	32	0.115 736 475 053 40 $\times 10^{-6}$
4	0	6	-0.584 576 160 480 39$\times 10^{1}$	14	2	10	-0.258 586 412 820 73 $\times 10^{-4}$
5	0	22	-0.152 854 824 131 40$\times 10^{-3}$	15	2	32	-0.406 443 630 847 99 $\times 10^{-8}$
6	0	32	-0.108 667 076 953 77$\times 10^{-5}$	16	3	10	0.664 561 861 916 35 $\times 10^{-7}$
7	1	0	-0.133 917 448 726 02$\times 10^{2}$	17	3	32	0.806 707 341 030 27 $\times 10^{-10}$
8	1	1	0.432 110 391 835 59 $\times 10^{2}$	18	4	32	-0.934 777 712 139 47 $\times 10^{-12}$
9	1	2	-0.540 100 671 705 06$\times 10^{2}$	19	5	32	0.582 654 420 206 01 $\times 10^{-14}$
10	1	3	0.305 358 922 039 16 $\times 10^{2}$	20	6	32	-0.150 201 859 535 03 $\times 10^{-16}$

Tabulka 5.4: Tabulka koeficientu bezrozměrné zpětné rovnice $T(p,h)$ pro oblast 1 [10].

Bezrozměrná forma rovnice $T(p,s)$ pro oblast 1 pak vypadá následovně:

$$\frac{T(p,s)}{T^*} = \theta(\pi\sigma) = \sum_{i=1}^{20} n_i \pi^{I_i} (\sigma + 2)^{J_i} \tag{5.6}$$

kde $\theta = T/T^*$, $\pi = p/p^*$ a $\sigma = s/s^*$ s $T^* = 1\ K$, $p^* = 1\ MPa$ a $s^* = 1\ Kj\ kg^{-1}\ K^{-1}$ [10].

i	I_i	J_i	n_i	i	I_i	J_i	n_i
1	0	0	0.174 782 680 583 07$\times 10^{3}$	11	1	12	0.356 721 106 073 66$\times 10^{-9}$
2	0	1	0.348 069 308 928 73$\times 10^{2}$	12	1	31	0.173 324 969 948 95$\times 10^{-23}$
3	0	2	0.652 925 849 784 55$\times 10^{1}$	13	2	0	0.566 089 006 548 37$\times 10^{-3}$
4	0	3	0.330 399 817 754 89	14	2	1	– 0.326 354 831 397 17$\times 10^{-3}$
5	0	11	– 0.192 813 829 231 96 $\times 10^{-6}$	15	2	2	0.447 782 866 906 32$\times 10^{-4}$
6	0	31	– 0.249 091 972 445 73$\times 10^{-22}$	16	2	9	– 0.513 221 569 085 07$\times 10^{-9}$
7	1	0	0.261 076 364 893 32	17	2	31	– 0.425 226 570 422 07$\times 10^{-25}$
8	1	1	0.225 929 659 815 86	18	3	10	0.264 004 413 606 89$\times 10^{-12}$
9	1	2	– 0.642 564 633 952 26$\times 10^{-1}$	19	3	32	0.781 246 004 597 23$\times 10^{-28}$
10	1	3	0.788 762 892 705 26 $\times 10^{-2}$	20	4	32	– 0.307 321 999 036 68$\times 10^{-30}$

Tabulka 5.5: Tabulka koeficientu bezrozměrné zpětné rovnice $T(p,s)$ pro oblast 1 [10].

6. XSteam

Zadáním práce je vytvořit scripty pro tvorbu h-s a t-s diagramů vodní páry a program pro řešení dějů ve vodní páře v prostředí Matlab za pomoci knihovny XSteam. Jedním z klíčových pojmů je zde knihovna XSteam. Knihovna je dostupná ve dvou verzích, pro MS Excel respektive OpenOffice a pro Matlab. Zvolené vývojové prostředí je Matlab a tedy zde bude podrobněji rozebrána pouze verze knihovny XSteam pro toto vývojové prostředí.

6.1 Licence

Licence nám umožňuje pro účely této práce knihovnu použít za určitých podmínek. Podmínkami je dodržení copyrightu, uvedení disclaimeru, přiložení disclaimeru k libovolnému produktu obsahující knihovnu XSteam a nesmíme zneužít jméno autora knihovny ani jeho webovou adresu bez jeho svolení [11].

Soubor license.txt obsahující v copyright a disclaimer je přiložen ke knihovně. Soubor se nachází uvnitř složky XSteam uvnitř všech scriptů a programů vytvořených pro účely této práce a to včetně všech verzí během vývoje. Tedy i kdyby došlo k omylnému zveřejnění libovolné hotové nebo i nehotové verze scriptu nebo programu, tak je obsahem soubor license.txt. Tedy je obsahem copyright i disclaimer.

6.2 O XSteam-u

XSteam je implementací standardu IAPWS IF97 [12]. Tato implementace umožňuje výpočty veličin uvedených v následující tabulce.

Notation	Quantity	Unit
T	Temperature	$°C$
p	Pressure	bar
h	Enthalpy	$kJ \cdot kg^{-1}$
v	Specific volume	$m^3 \cdot kg^{-1}$
rho	Density	$kg \cdot m^{-3}$
s	Specific entropy	$kJ \cdot kg^{-1} \cdot °C^{-1}$
u	Specific internal energy	$kJ \cdot kg^{-1}$
C_p	Specific isobaric heat capacity	$kJ \cdot kg^{-1} \cdot °C^{-1}$
C_v	Specific isochoric heat capacity	$kJ \cdot kg^{-1} \cdot °C^{-1}$
w	Speed of sound	$m \cdot s^{-1}$
my	Viscosity	$Pa \cdot s$
tc	Thermal conductivity	$W \cdot m^{-1} \cdot degreeC^{-1}$
st	Surface tension	$N \cdot m^{-1}$
x	Vapour fraction (0-1)	–
vx	Vapour volume fraction (0-1)	–

Tabulka 6.1: Notace a jednotky knihovny XSteam [12]

Z tabulky 6.1 vyplývá, že XSteam pracuje v určitých měrných jednotkách. Je tedy zapotřebí zajistit, aby funkce knihovny XSteam vždy byly volány s hodnotami ve správných jednotkách.

Jedná se o implementaci pro prostředí Matlab, tedy přirozeně používá syntax tohoto prostředí. Obecný příklad volání funkce je:

```
Výstup = XSteam('název funkce', vstup1, vstup2),
```

Konkrétně pak:

```
t = XSteam('T\_ph',p,h);
```

nebo případně:

```
t = XSteam('Tsat\_p',p);
```

Počet vstupů (tj. jeden nebo dva) je určený názvem funkce. Dostupné názvy funkcí a tedy i příslušné výpočty jsou uvedené v tabulce 6.2.

Název funkce	Vstup 1	Vstup 2	výstup
Tsat_p	p		Saturační teplota jako funkce tlaku
T_ph	p	h	Teplota jako funkce tlaku a entalpie
T_ps	p	s	Teplota jako funkce tlaku a entropie
T_hs	h	s	Teplota jako funkce entalpie a entropie
psat_T	T		Saturační tlak jako funkce teploty
p_hs	h	s	Tlak jako funkce entalpie a entropie
p_hrho	h	rho	Tlak jako funkce entalpie a hustoty. Nepřesné v oblasti vody.
hV_p	p		Entalpie na mezi saturace páry (x = 1) jako funkce tlaku
hL_p	p		Entalpie na mezi saturace vody (x = 0) jako funkce tlaku
hV_T	t		Entalpie na mezi saturace páry (x = 1) jako funkce teploty
hL_T	t		Entalpie na mezi saturace vody (x = 0) jako funkce teploty
h_pT	p	t	Entalpie jako funkce tlaku a teploty
h_ps	p	s	Entalpie jako funkce tlaku a entropie
h_px	p	x	Entalpie jako funkce tlaku suchosti
h_Tx	t	x	Entalpie jako funkce teploty a suchosti
h_prho	p	rho	Entalpie jako funkce tlaku a hustoty
vV_p	p		Objem na mezi saturace páry (x = 1) jako funkce tlaku
vL_p	p		Objem na mezi saturace vody (x = 0) jako funkce tlaku
vV_T	t		Objem na mezi saturace páry (x = 1) jako funkce teploty
vL_T	t		Objem na mezi saturace vody (x = 0) jako funkce teploty
v_pT	p	t	Měrný objem jako funkce tlaku a teploty
v_ph	p	h	Měrný objem jako funkce tlaku a entalpie
v_ps	p	s	Měrný objem jako funkce tlaku a entropie
rhoV_p	p		Hustota na mezi saturace páry (x = 1) jako funkce tlaku
rhoL_p	p		Hustota na mezi saturace vody (x = 0) jako funkce tlaku
rhoV_t	t		Hustota na mezi saturace páry (x = 1) jako funkce teploty
rhoV_t	t		Hustota na mezi saturace vody (x = 0) jako funkce teploty
rho_pT	p	t	Hustota jako funkce tlaku a teploty
rho_ph	p	h	Hustota jako funkce tlaku a entalpie
rho_ps	p	s	Hustota jako funkce tlaku a entropie

Název funkce	Vstup 1	Vstup 2	výstup
sV_p	p		Entropie na mezi saturace páry (x = 1) jako funkce tlaku
sL_p	p		Entropie na mezi saturace vody (x = 0) jako funkce tlaku
sP_t	t		Entropie na mezi saturace páry (x = 1) jako funkce teploty
sL_t	t		Entropie na mezi saturace vody (x = 0) jako funkce teploty
s_pT	p	t	Měrná entropie jako funkce tlaku a teploty
s_ph	p	h	Měrná entropie jako funkce tlaku a entalpie
uV_p	p		Vnitřní energie na mezi saturace páry (x = 1) jako funkce tlaku
uL_p	p		Vnitřní energie na mezi saturace vody (x = 0) jako funkce tlaku
uV_T	t		Vnitřní energie na mezi saturace páry (x = 1) jako funkce teploty
uL_T	t		Vnitřní energie na mezi saturace vody (x = 0) jako funkce teploty
u_pT	p	t	Vnitřní energie jako funkce tlaku a teploty
u_ph	p	h	Vnitřní energie jako funkce tlaku a entalpie
u_ps	p	s	Vnitřní energie jako funkce tlaku a entropie
CpV_p	p		Tepelná kapacita (p) na mezi saturace páry (x = 1) jako funkce tlaku
CpL_p	p		Tepelná kapacita (p) na mezi saturace páry (x = 0) jako funkce tlaku
CpV_T	t		Tepelná kapacita (p) na mezi saturace páry (x = 1) jako funkce teploty
CpL_T	t		Tepelná kapacita (p) na mezi saturace páry (x = 0) jako funkce teploty
Cp_pT	p	t	Tepelná kapacita (p) jako funkce tlaku a teploty
Cp_ph	p	h	Tepelná kapacita (p) jako funkce tlaku a entalpie
Cp_ps	p	s	Tepelná kapacita (p) jako funkce tlaku a entropie
CvV_p	p		Tepelná kapacita (v) na mezi saturace páry (x = 1) jako funkce tlaku
CcL_p	p		Tepelná kapacita (v) na mezi saturace páry (x = 0) jako funkce tlaku
CvV_T	t		Tepelná kapacita (v) na mezi saturace páry (x = 1) jako funkce teploty
CvL_T	t		Tepelná kapacita (v) na mezi saturace páry (x = 0) jako funkce teploty
Cv_pT	p	t	Tepelná kapacita (v) jako funkce tlaku a teploty
Cv_ph	p	h	Tepelná kapacita (v) jako funkce tlaku a entalpie
Cv_ps	p	s	Tepelná kapacita (v) jako funkce tlaku a entropie
wV_p	p		Rychlost zvuku na mezi saturace páry (x = 1) jako funkce tlaku
wL_p	p		Rychlost zvuku na mezi saturace páry (x = 0) jako funkce tlaku
wV_T	t		Rychlost zvuku na mezi saturace páry (x = 1) jako funkce teploty
wL_T	t		Rychlost zvuku na mezi saturace páry (x = 0) jako funkce teploty
w_pT	p	t	Rychlost zvuku jako funkce tlaku a teploty
w_ph	p	h	Rychlost zvuku jako funkce tlaku a entalpie
w_ps	p	s	Rychlost zvuku jako funkce tlaku a entropie

Tabulka 6.2: Funkce knihovny XSteam [12]. Začíná na předcházející straně.

Z tabulky 6.2 je zřejmé, že knihovna XSteam neumožňuje spočítat veškeré, pro tuto účely této práce nezbytné, kombinace vstupů. Například neumožňuje spočítat libovolné zadání pomocí objemu. Tyto nedostatky bude potřeba vyřešit vhodnou prací s touto knihovnou.

7. Provedení práce

V této kapitole je popsáno řešení zadání a uvedené příklady kódu, respektive útržky scriptu. Nejedná se o programovou dokumentaci. Ta je provedena formou komentářů uvnitř scriptu a funkcí. Následující text popisuje funkci skriptu pro vykreslení diagramů, strukturu programu a popis jeho nejdůležitějších funkci.

7.1 Diagramy

Prvním cílem diplomové práce bylo vytvořit h-s a t-s diagramy vodní páry. V době tvorby této práce byly k dispozici diagramy od autora ing. Hugo Šena [5]. Z těchto diagramů byla čerpána inspirace pro určení barevného rozložení izočar. Nově vytvořené diagramy, mimo všech příslušných náležitostí diagramu, obsahují štítek s typem diagramu (h-s respektive t-s diagram vody a vodní páry), informací o standardu dle kterého byly vytvořeny (IAPWS-IF97), nástrojem použitým pro vytvoření diagramů (XSteam 2.6), autorem, datem a místem.

Izočáry	Jednotky	h-s diagram	t-s diagram
		Barva	Barva
izotermy	$°C$	červená	-
izobary	MPa	šedá	červená
izochory	$m^3 \cdot kg^{-1}$	zelená	zelená
suchosti	$-$	modrá	šedá
izoentalpy	$kJ \cdot kg^{-1}$	-	modrá

Tabulka 7.1: Tabulka jednotek a barev izočar zobrazených v diagramech.

K vykreslení diagramů a k výpočtům bodů na jednotlivých izočárách bylo v zadaném prostředí Matlabu vytvořeno několik souborů s funkcemi, jeden soubor s konstantami a hlavní script. Jednotlivé typy souborů budou podrobně popsány v následujících kapitolách. Struktura tohoto projektu je pak následující. Soubory s funkcemi jsou rozděleny do dvou složek a složky jsou pojmenovány *popisky* a *vypoctove_plot*. V první zmíněné složce, popisky, jsou uloženy soubory s funkcemi generujícími popisky k izočarám v diagramech (např. 50 $°C$). Ve druhé zmíněné složce, vypoctove_plot jsou ostatní funkce scriptu. Níže jsou uvedené jednotlivé typy funkcí s příklady fungování.

7.1.1 Popisky

Během vývoje funkcí k vykreslení diagramů proběhlo několik pokusů o automatizaci vykreslování popisků jednotlivých izočar. Postupně bylo vytvořeno několik verzí automatického polohování a natáčení, a poté pouze automatického natáčení. Ani jedna z verzí neposkytovala výstup v požadovaném tvaru. Vždy některý popisek nebyl v poloze, ve které byl očekáván a často docházelo k překryvu popisků jednotlivých izočar. Proto byl zvolen postup zdlouhavého manuálního umisťování jednotlivých popisků. Pro každý z diagramů zvlášť má každý popisek ručně definované natočení a souřadnice v diagramu.

Popisky jsou pak rozdělené dle typu izočáry a diagramu do jednotlivých souborů. Soubory (m-soubory) jsou ve složce popisky, která je umístěná v hlavní složce tohoto projektu. Každý soubor je funkcí, která

nemá žádný vstup a vrací seznam struktur s popisky. Například ze souboru TS_isoh_popisky.m definice popisku izoentalpy 2500 $kJ \cdot kg^{-1}$ v t-s diagramu vypadá následovně:

```
p.a(i) = "2500"; % hodnota izoentalpy
p.t(i) = 30;     % y-ová souřadnice
p.s(i) = 8.4;    % x-ová souřadnice
p.r(i) = -44;    % natočení ve stupních
```

Kde p.a je popisek ve formátu string, p.t a p.s jsou polohy na osách t-s diagramu a p.r je natočení popisku. Pro úplnost i značí index tohoto popisku v seznamu (listu) p.

7.1.2 Konstanty

Vykreslování diagramů probíhá pomocí hodnot, jež se během jednoho vykreslení diagramů nemění. Například každý diagram zobrazuje určitou oblast, která byla definována pomocí konstant minimální a maximální hodnoty na jednotlivých osách. Druhým příkladem je dříve zmíněné určení barev jednotlivých izočar. Dále jsou ve výpočtech zahrnuté fyzikální konstanty. Jejich definice (umístění) je pro snadný náhled sjednocena s ostatními konstantami skriptu.

Konstanty pro vykreslení diagramů jsou umístěny ve složce vypoctove_plot v souboru initKonst.m. Jedná se o funkci prostředí Matlabu bez vstupních hodnot se třemi výstupními strukturami [konstHS, konstTS, barva] = initKonst(). První z výstupních struktur obsahuje konstanty definující h-s diagram.

Struktura konstHS obsahuje parametry mezních hodnot na osách x a y, respektive s a h. Dále struktura obsahuje definice počtu bodů většiny izočar. V některých případech bylo nutné výpočet bodů izočáry rozdělit na úseky. Příkladem je výpočet izotermy v h-s diagramu. Izoterma v h-s diagramu byla počítána především za pomoci funkcí h_pT a s_pT. Tyto funkce jsou vzhledem ke skutečnosti, že izotermy a izobary v oblasti mokré páry splývají, použitelné pouze v oblasti syté páry. Tedy je nutné pro oblast mokré páry použít jinou kombinaci funkcí XSteam-u. Byly zvoleny funkce h_Tx a s_pH. Tím vznikají dva intervaly, ve kterých je vhodné definovat počty výpočetních bodů. V souboru initKonst jsou počty výpočetních bodů nazývány rozlišením, tj. například rozliseni_isotermy_suche udává počet bodů na izotermě v oblasti suché páry. Dalším objektem struktury je položka tlaku. Jedná se o list výpočtových bodů pro mezní křivky $x = 0$ a $x = 1$. Struktura konstTS je velmi podobná struktuře konstHS s tím, že v t-s diagramu není nutné vykreslit izotermy, ale navíc jsou vykresleny izoentalpy. Absence izoterem znamená odstranění s nimi spojených nastavení jako jsou rozlišení izoterem v oblastech mokré a suché páry. Izoentalpy v t-s diagramu podobná nastavení nemají.

Poslední strukturou souboru initKonst.m je definice tlouštěk izočar a barevných rozložení diagramů. Izočáry jsou rozděleny do dvou skupin, hlavní a vedlejší. Hlavní izočáry jsou takové, které mají větší význam. Například v h-s diagramu je za hlavní izotermu považováno každých 50°C, tj. 50, 100, 150°C..., a za vedlejší izotermy jsou považovány hodnoty mezi, tj. 60, 70, 80, 90, 110°C. Optické rozdělení hlavních a vedlejších izočar je dvojí. Tloušťka čáry a barva, respektive odstín barvy. Hlavní izočáry mají být výraznější, tedy jsou tlustší (1,2 bodu) a vedlejší jsou tenší (0,6 bodu). Odstín barvy je pak tmavší pro hlavní izočáru a světleji pro vedlejší izočáru.

7.1.3 Paralist

Velkou nevýhodou knihovny XSteam pro projekty jako je vykreslení izočar je, že knihovna pracuje pouze s jedním vstupem. Není možné přímo použít funkce knihovny na celé seznamy, nebo matice hodnot. Je tedy na místě využít již existující, nebo vytvořit obal k této knihovně. Tento obal by vzal daný soubor hodnot a po jedné je vkládal jako vstupní parametry pro funkce XSteam-u a následně výstupní hodnoty XSteam-u by ukládal do příslušné struktury hodnot.

Jednou možností je použít existující knihovnu XSteamW [13]. Tato knihovna je umožňuje volat XSteam jako funkci dvou listů a vrací matici o rozměrech m, n, kde m je délka prvního listu (vektoru) a n je délka druhého vstupního listu (vektoru). Tato vlastnost vede k různému výstupu v případě zadání list,

hodnota a hodnota, list. Toto chování by vzhledem k chování funkce plot() Matlabu znamenalo nutnost při každém volání XSteamW kontrolovat orientaci výstupního vektoru a případně vektor otáčet.

Vzhledem k obtížnosti výroby tohoto obalu, která pro potřeby tohoto scriptu (respektive programu) není vysoká, bylo rozhodnuto o jeho vytvoření. Funkce byla nazvána paralist a jedná se o zjednodušenou verzi XSteamW. Možnými vstupy jsou pouze list a jedna hodnota, výstup je pak list (vektor) vždy se správným rozměrem. List samozřejmě může být degenerovaný na jednu hodnotu.

7.1.4 Vykreslovací funkce

Dále byly pro účely vykreslení diagramů vytvořené funkce HS_plot a TS_plot. Jedná se o dvojici funkcí, které vytváří samotné diagramy. Na vstupu očekávají konstanty příslušných diagramů, h-s respektive t-s, strukturu konstant barev a několik 'y' (vykresli) nebo 'n' (nevykresli) pro popisky a příslušné izočáry. Například pro h-s diagram funkce vypadá následovně

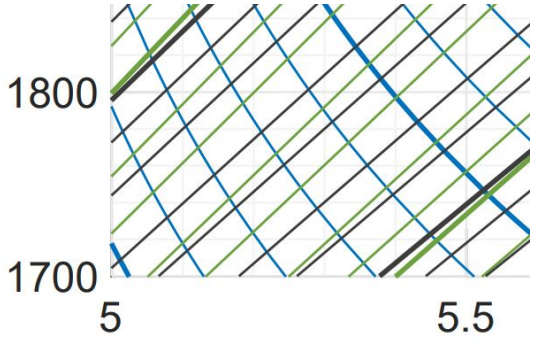
```
HS_diagram = HS_plot(konst, barva, popisky, izobary, izotermy, suchosti, izochory),
```

respektive

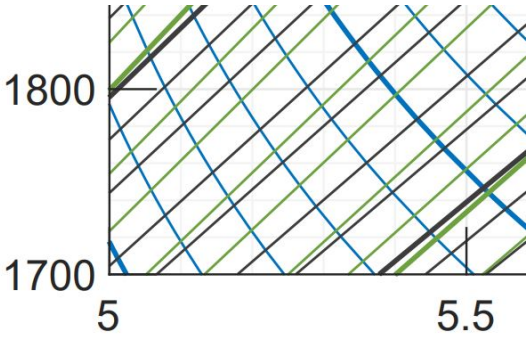
```
HS_diagram = HS_plot(konstHS, barvy, 'y', 'y', 'y', 'y', 'y').
```

Následuje popis funkce HS_plot, funkce TS_plot je její analogií a nebude zde vysvětlena. Funkce HS_plot vytvoří figuru HS_diagram, definuje rozměry figury a okraje. Dále je nastaven renderer na painters [14] [15] (*set(gcf,'renderer','painters')*, kde gcf je pojem Matlabu odkazující na aktuální aktivní figuru, v tomto případě h-s diagram).

Funkce HS_plot dále definuje osy, značky na osách a vrstvu ve které jsou osy diagramu zobrazeny. Osy jsou vykresleny ve vrchní vrstvě. To má za následek překrytí krajních bodů izočar a ve výsledku diagramy vypadají čistěji. Chování vykreslení je znázorněno na 7.1 a 7.2.



Obrázek 7.1: Osy ponechané v základním nastavení



Obrázek 7.2: Osy ve vrchní vrstvě

Funkce HS_plot provádí dva další úkony. První z nich je nastavení barev a tlouštěk linií jednotlivých izočar. Druhým z nich je vykreslení jednotlivých izočar. Jednotlivé izočáry jsou definovány v listech izočar k vykreslení. Tyto listy jsou dále děleny na listy hlavních izočar a listy vedlejších izočar. Listy jsou poté vloženy jako vstupní hodnoty *for* cyklů. Pro názornost jsou zde zobrazeny izobary. Prvně jsou definovány hlavní izobary:

```
isobaryM = [100, 10, 1, .1, .01, .001]*10;
```

Dále jsou definovány vedlejší izobary:

```
isobary\_p = [100, 80, 60, ..., 0.003, 0.002, 0.001]*10;
```

Následují for cykly výpočtů bodů a vykreslení izočar. Uvnitř cyklů je volána výpočetní funkce *vypocetHS*, která je podrobně popsána v nadcházející sekci. Následně je volána funkce Matlabu plot s příslušnými parametry:

```
for i = isobary_p
    [malujH,malujS] = vypocetHS(konst,'isobara',i);
    plot(malujS,malujH,'Color',barva.hs_p_H,'Linewidth',...
        barva.tloustka_V)
end
for i = isobaryM
    [malujH,malujS] = vypocetHS(konst,'isobara',i);
    plot(malujS,malujH,'Color',barva.hs_p_H,'Linewidth',...
        barva.tloustka_H)
end
```

Na výše uvedeném příkladu definice izobar k vykreslení, je možné si povšimnout problematiky převodu jednotek "na jednotky XSteam-u". Konkrétně tedy poslední část definice listu izobar $*10$. Definujeme izobary $100MPa$, ale jak bylo zmíněno dříve v této práci, XSteam očekává vstup v jednotce *bar*. Tedy musíme provést převod jednotek a hodnotu 100 vynásobit deseti.

7.1.5 Výpočty diagramů

K vykreslení izočáry je zapotřebí znát body, které na ní leží. To zařizují dva listy hodnot, jeden pro osu x, druhý pro osu y. Respektive dle příkladu h-s diagramu pro osy s (osa x) a h (osa y). Hodnoty uložené v těchto listech jsou spočítány a následně předány k dalšímu zpracování pomocí funkce vypocetHS. Tato funkce požaduje jako vstup dříve zmíněnou strukturu příslušných konstant, v tomto případě h-s konstant, string (text) s názvem izočáry a hodnotu této izočáry. Příklad volání této funkce je:

```
[outH, outS] = vypocetHS(konstHS,'isobara',50);
```

Funkce vrátí listy hodnot h a s pro izobaru $50bar$. Funkce vypocetHS volá další výpočetní funkce, jako je dříve zmíněná funkce *paralist*. Případné rozdělení izočáry dle oblastí diagramu je provedeno v rámci této funkce.

Výpočty jsou, kde je to možné, provedené pomocí funkcí knihovny XSteam tak, že hodnota izočáry je jedním ze zadávaných parametrů - h nebo s je druhým zadávaným parametrem a výstup je třetí požadovanou hodnotou. Pokud tento elegantní výpočet není možný, tak přichází na řadu čtvrtá veličina, pomocí které získáme alespoň jeden z požadovaných výstupů. Druhý požadovaný výstup pak již lze snadno dopočítat. Pro vysvětlení je zde uveden příklad. Řekněme, že je cílem vykreslit (spočítat) izobaru. Podíváme se do tabulky funkcí knihovny XSteam 6.2 a zjistíme, že XSteam má funkci h_ps. Protože známe hodnotu izobary kterou chceme spočítat, tak známe tlak. Tedy stačí si vytvořit seznam hodnot entropií ve kterých chceme výpočetní body a ty dosadit do funkce h_ps.

Ostatní izočáry v h-s diagramu jsou spočítány podobným způsobem. Hodnoty pro t-s diagram jsou vypočítané podobným postupem pomocí funkce vypocetTS.

7.1.6 Izochory

Výpočet bodů izochor nen í triviální a je v této kapitole vysvětlen. Dle tabulky 6.2 knihovna XSteam obsahuje funkce pro výpočet objemu na mezi suchosti ($x = 0$, $x = 1$) vV_p, vL_p, vV_T, vL_T a funkce v_pt, v_ph a v_ps. Funkce pro výpočet objemu na mezi suchosti umožňuje na dané izochoře pouze jeden bod. To pro vykreslení izochory v celé oblasti h-s a t-s diagramů nestačí. Tedy jsou pro účely vykreslení diagramů (izochor) vhodně využité funkce funkce v_pt, v_ph a v_ps.

Je znám objem, který je žádoucí dopočítat ze vstupů tlak - teplota, entalpie nebo entropie. Tedy je nalezena taková kombinaci vstupů, která na výstupu příslušné XSteam funkce vrátí správný objem.

Zároveň vzájemné rozložení těchto kombinací zadání musí být přiměřené pro vykreslení izochory v h-s a t-s diagramech. Požadavek rovnoměrného rozložení bodů je možné splnit fixací jedné ze vstupních proměnných a to s přiměřenými rozestupy. Druhá veličina je pak zadána tak, aby se výstup funkce XSteam-u shodoval s požadovaným objemem (se zadanou izochorou).

Tedy jedna ze vstupních veličin je fixována a druhou je nutné nalézt. Ta je nalezena pomocí iterační metody. Výchozí hodnotou je číslo, se kterým XSteam zaručeně umí počítat. Následně je pak iterováno, dokud se s výstupní hodnota z příslušných funkcí XSteam nepřiblíží dostatečně blízko k hodnotě vykreslované izochory.

Samotný výpočet izochor je proveden pomocí funkce isochory_vypocet:

```
[out_s, out_h, out_T] = isochory_vypocet(vol, varargin),
```

kde vol je hodnota vykreslované izochory a varargin je volitelný vstup, definující zdali se jedná o výpočet pro h-s nebo t-s diagram. Pokud je varagin roven stringu hs, tak je výpočet hodnot (rozsah) upraven pro h-s diagram. Pokud tento volitelný vstup chybí, tak je výpočet proveden s větším rozsahem mezí, tedy pro t-s diagram.

Funkce dělí výpočet na dvě oblasti, na oblast mokré páry a na oblast suché páry. Dělení probíhá pomocí hodnoty tlaku na mezi suchosti. Ten je prvně spočítán jako teplota na mezi suchosti a následně převeden pomocí funkce psat_T.

```
pmez = XSteam('psat_T', temperature);
```

Než je ale možné provést tento převod, tak je nutné mít co převést. Tomuto příkazu tedy předchází výpočet teploty na mezi suchosti ($x = 1$) pro daný objem. To je provedeno iteračně tak, že z teploty $1°C$ je postupně krokováno směrem k vyšším teplotám. A to dokud hodnota objemu vypočteného pomocí funkce vV_T není větší než hodnota objemu zadané izochory. V okamžiku, kdy je vypočtená hodnota větší, udělá algoritmus krok zpět a zmenší krok na polovinu. Dále je implementovaná ochrana proti překročení teploty kritického bodu $373.94°C$. Tato teplota byla získána ručním opakovaným voláním funkce vV_T. Teplota byla následně ověřena v literatuře. Literatura udává teplotu kritického bodu $373.95°C$ [3]. Vzhledem k použité knihovně XSteam a jejím omezením je vhodné určit teplotu pro kterou vV_T vrací objem a která je v blízkosti kritického bodu. Výsledný kousek kódu pak vypadá následovně:

```
while (dist > citlivost && stepT > .00000001)
    holdit = XSteam('vV_T', temperature);
    dist = vol - holdit;
    if dist < 0
        temperature = temperature + stepT;
    elseif dist > 0
        temperature = temperature - stepT;
        stepT = stepT * .5;
    elseif dist == 0 % continue
    end
    if temperature > 373.94
        temperature = 373.94;
        citac = citac + 1;
        if citac > 5
            error("dosazena maximalni teplota")
        end
    end
    dist = abs(dist);
end
```

Následuje dříve zmíněný převod teploty na tlak, tvorba listu hodnot tlaku a obdoba výše uvedeného kódu pro oblast mokré páry zabalená ve for cyklu pro každou hodnotu v listu. V této oblasti je veličinou

se zvoleným rozestupem pro vykreslení izočáry tlak. Po definici listu hodnot tlaků následuje ve for cyklu zabalená obdoba hledání objemu na mezní křivce. While cyklus je stejný s tím rozdílem, že místo funkce vV_T je volána funkce v_ps a není iterována hodnota teploty ale entropie. Po nalezení hodnoty zmíněným while cyklem je známa hodnota objemu, tlaku a entropie. Pro vykreslení h-s a t-s diagramů je ještě zapotřebí znát entalpii a teplotu. Tyto hodnoty jsou dopočítané pomocí funkcí T_ps a h_ps. Pak již jen dojde k uložení dopočítaných hodnot entropie, entalpie a teploty do jejich příslušných listů.

Zbývá dopočítat hodnoty v oblasti suché páry. V oblasti suché páry je využita funkce v_pT. Pro výpočty v této oblasti byla fixována teplota a iterační veličinou je tlak. Dolní mez teploty je spočítána pomocí mezní křivky a maximální teplota $800°C$ je dána zobrazovanou oblastí diagramů. Tedy je vytvořen list hodnot teplot v daných mezích. Po definici listu teplot následuje obdoba for cyklu výpočtu v oblasti mokré páry, a to včetně vnořeného while cyklu. Rozdílem je využití jiné funkce XSteam. Místo funkce v_ps je využitou funkcí v_pT.

Téměř posledním úkonem funkce isochory_vypocet je spočítané hodnoty jednotlivých oblastí spojit dohromady tak, aby výstupní listy byly plynulé. Tím je na mysli například to ,že hodnoty teploty musejí být seřazené od nejmenší po největší nebo naopak. Co není žádoucí je, aby na spojení listů jednotlivých oblastí došlo ke skoku. Například není vhodné, aby teploty nejdříve klesaly od teploty na mezi suchosti k nule, a v místě spojení došlo z téměř nulové hodnoty ke skoku na teplotu opět na mezi suchosti. Takovýto nevhodný list pak ve funkci Matlabu plot dělá skoky a nežádané čáry. Příklad řešení je níže.

```
out_s = [flip(s_lower), s_higher];
out_h = [flip(h_lower), h_higher];
out_T = [flip(t_lower), t_higher];
```

Přímým vykreslením touto funkcí vytvořených listů docházelo k vykreslení nedokonalostí v okolí meze suchosti ($x = 1$). Tedy byl implementován filtr hodnot kontrolující velikost hodnoty entropie. Pokud je ve výstupním listu skok na nižší hodnotu, tak je hodnota označena jako "k odstranění". Jinými slovy kontroluji monotónnost listu.

K odstranění označených hodnot, pokud nějaké jsou, slouží další cyklus. Protože funkce Matlabu plot vyžaduje vstup kompatibilních rozměrů, tak jsou odstraněny hodnoty na označené pozici listu entropie ze všech výstupních listů. Řešení filtru je ukázáno níže:

```
for s = (1:length(out_s))
    if out_s(s) > last_s
        last_s = out_s(s);
    else
        rm_these(end+1) = s;
    end
end
if ~(isempty(rm_these))
    for rm = (1:length(rm_these))
        out_s = out_s([1:rm_these(length(rm_these)-rm+1)-1,...
            rm_these(length(rm_these)-rm+1)+1:end]);
        out_h = out_h([1:rm_these(length(rm_these)-rm+1)-1,...
            rm_these(length(rm_these)-rm+1)+1:end]);
        out_T = out_T([1:rm_these(length(rm_these)-rm+1)-1,...
            rm_these(length(rm_these)-rm+1)+1:end]);
    end
end
```

Z výše uvedeného postupu vyplývá omezení objemů, které je možné za pomoci této funkce vykreslit. Specificky omezení mezní křivkou, respektive teplotou kritického bodu. Funkce isochory_vypocet předpokládá rozdělení izochory na dvě části pomocí mezní křivky. Pokud izochora nemá průsečík s mezní

křivkou, tak není možné ji rozdělit do oblasti mokré a suché páry a tudíž pro její vykreslení není možné využít této funkce. V předešlém textu byla maximální teplota určena na $373,94\ °C$. Zavoláním funkce vV_T s touto teplotou získáme maximální objem, se kterým tato funkce může pracovat. Hodnota tohoto objemu je $220,6239\ m^3 \cdot kg^{-1}$.

7.1.7 Výsledné diagramy

Zadáním bylo v prostředí Matlab vytvořit script pro vykreslení h-s a t-s diagramů. Byl vytvořen script hs_ts_generator.m. Jeho spuštěním jsou postupně vykreslovány oba diagramy. První vykreslený diagram je h-s. Ten je scriptem vykreslen tak, že je ze scriptu hs_ts_generator zavolána funkce HS_plot. Uvnitř této funkce jsou definovány parametry h-s diagramu. Plocha diagramu je pak vyplněna jednotlivými izočárami. Ty jsou definovány pomocí listů hodnot. Pro každou z hodnot v těchto listech je pak zavolána funkce vypocetHS, která spočítá jednotlivé body na izočáře a ty poté vrátí ve formě listu. Listy navrácených hodnot jsou dva, jeden pro hodnoty na ose s, druhý pro hodnoty na ose h. Tyto listy jsou poté vloženy do funkce Matlabu plot s příslušnými parametry upravujícími vzhled izočáry. Po vykreslení všech izočar jednoho typu, například po vykreslení všech izochor následuje vykreslení všech izoterem. Po vykreslení všech izočar je do diagramu vložen štítek s logem a informacemi o diagramu. Tím je vykreslení h-s diagramu ukončeno a figura je předána zpět do hlavního scriptu.

Hlavní script figuru uloží do složky scriptu jako outHS.pdf a spustí vykreslení t-s diagramu. Vykreslení t-s diagramu probíhá pomocí funkce TS_plot analogicky k vykreslení h-s diagramu. Po navrácení figury t-s diagramu do hlavního scriptu je figura opět uložena do složky scriptu jako outTS.pdf. Tím je běh scriptu pro vykreslení h-s a t-s diagramu ukončen.

Diagramy jsou umístěné v příloze této práce. Diagramy byly vytvořeny pro tisk na papír formátu A3. Tedy v této práci tištěné na formát papíru A4 nemusejí být diagramy tak přehledné, jako na papíře pro nějž byly diagramy vytvořeny. Například popisky jsou menší a hůře čitelné.

7.2 Program

Druhým cílem této práce bylo vytvořit program pro řešení dějů ve vodní páře. Kd osažení cíle byl tento úkol rozdělen do několika menších kroků.

K vytvoření programu je zapotřebí sestavit funkce, které ze dvou zadaných termodynamických parametrů dopočítají parametry ostatní. Jinak řečeno, jsou zapotřebí výpočetní funkce. Dále program musí mít uživatelské rozhraní, jinak se nejedná o program ale o skript. V rámci uživatelského prostředí by program měl zadaný děj vykreslit.

Toto jsou tři pod-úkoly které vyplývají ze zadání. V této sekci práce je popsáno řešení těchto úkolů pomocí různých funkcí včetně jejich vnitřní logiky a fungování.

7.2.1 Uživatelské rozhraní

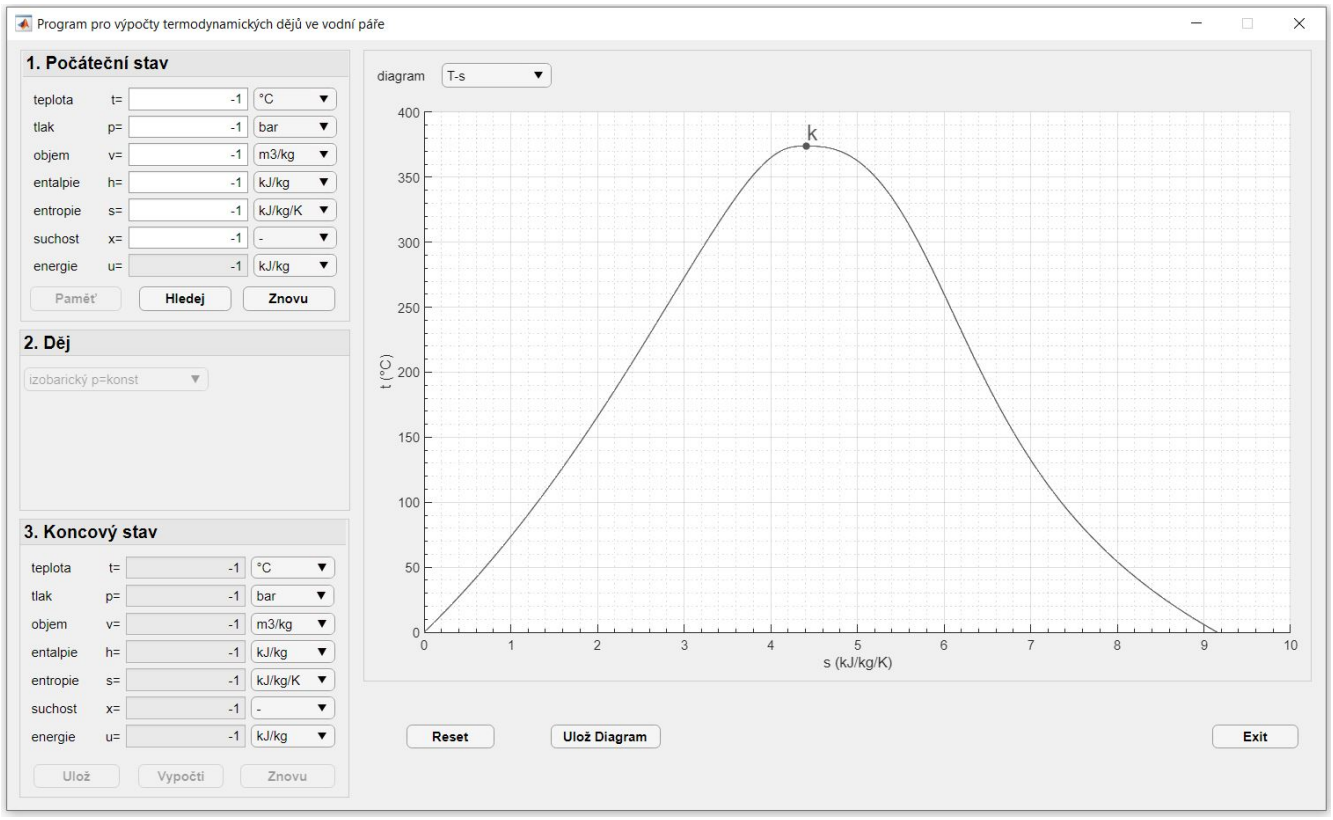
Hlavní okno uživatelského rozhraní programu bylo vytvořeno pomocí prostředí App Designer. V tomto prostředí bylo vytvořeno okno aplikace. Velikost tohoto okna byla z důvodů kompatibility s různými zařízeními zvolena 1200 x 700 bodů. Do tohoto okna byly poté vloženy prvky, oblast grafu a kontejnery (*panel*) s dalšími prvky. Největší část okna programu zabírá právě graf (diagram). Menší, levou část okna programu zabírá sloupec o třech kontejnerech především s termodynamickými veličinami.

V horním kontejneru, pojmenovaném *Počáteční stav*, jsou editovatelná pole termodynamických veličin t, p, v, h, s, x a needitovatelné pole vnitřní energie u. Pod těmito poli jsou, stále v rámci kontejneru, vložená tři tlačítka *Paměť*, *Hledej* a *Znovu*. Kliknutí na tlačítko *Paměť* otevře okno paměti termodynamických stavů a dějů. Toto okno je podrobně popsáno v sekci Paměť. Kliknutí na tlačítko *Hledej* vypočítá ze zadaných hodnot termodynamických veličin hodnoty ostatních veličin. Kliknutí na tlačítko *Znovu* vymaže hodnoty z políček *počátečního stavu* a také z políček *koncového stavu* a umožní tím zadání nového počátečního stavu (respektive celého děje).

Kontejner *Děj* zprvu obsahuje viditelnou položku "dropdown menu", s výběrem termodynamických dějů. V případě výběru adiabatického děje z dropdown menu se v kontejneru zobrazí editovatelné pole pro účinnost. Dále kontejner obsahuje pole s parametry děje, která se zobrazí až po jeho vypočtení.

Dolní kontejner *Koncový stav* je obdobou kontejneru počátečního stavu. Také obsahuje editační pole, odpovídající jednotlivým termodynamickým veličinám, pod kterými jsou tři tlačítka, *Ulož*, *Vypočti* a *Znovu*. Kliknutím na tlačítko *Vypočti* jsou pomocí počátečního stavu, zadaného děje a jedné veličiny koncového stavu dopočítány ostatní stavové veličiny koncového stavu. Kliknutí na tlačítko *Znovu* resetuje koncový stav. Klinutí na tlačítko *Ulož*, uloží vypočtený děj do paměti.

Okno programu mimo výše uvedených prvků také obsahuje další tři tlačítka pod diagramem, *Reset*, *Ulož Diagram* a *Exit*. Kliknutí na tlačítko *Reset* obnoví program do stavu v okamžiku po spuštění. Kliknutí na tlačítko *Ulož Diagram* otevře dialogové okno pro uložení zobrazeného digramu ve formátu pdf. Jedná se o standardní dialogové okno výběru lokace a názvu uložení souboru, respektive v tomto případě vykresleného diagramu. Tlačítko *Exit* otevře dialogové okno potvrzení ukončení programu.



Obrázek 7.3: Grafické provedení okna programu ve stavu po spuštění.

Mimo hlavního okna program obsahuje také několik vyskakovacích oken, některé z nich již byly zmíněny. Jejich funkcí je uživatele upozornit na chybu, předat mu doplňkovou informaci nebo požádat o potvrzení akce. Tyto funkce jsou vytvořené pomocí příkazů Matlabu jako je *warndlg*().

7.2.2 Funkce pro dopočítání neznámých parametrů

V úvodu sekce program bylo stanoveno, že je zapotřebí mít hotové funkce, které umožňují z kombinace dvou zadaných termodynamických veličin dopočítat ostatní termodynamické veličiny. Některé z těchto funkcí jsou k dispozici prostřednictvím knihovny XSteam. Funkce poskytnuté touto knihovnou jsou vypsány v tabulce 6.2. Možnosti knihovny XSteam pro účely této práce nejsou dostačující. Knihovna neobsahuje některé kombinace vstupních parametrů, například funkci pro výpočet tlaku zadanou pomocí suchosti a objemu. Tyto funkce byly doplněny iteračními funkcemi založenými na knihovně XSteam:

1. dopocet_h_pu
2. dopocet_h_pv
3. dopocet_p_ht
4. dopocet_p_hv
5. dopocet_p_hx
6. dopocet_p_st
7. dopocet_p_su
8. dopocet_p_sv
9. dopocet_p_sx
10. dopocet_p_tu
11. dopocet_p_tv
12. dopocet_p_ux
13. dopocet_p_vu
14. dopocet_p_vx

K těmto funkcím je ještě přiřazena pomocná výpočetní funkce *dopocet_pomoc_p*, která obsahuje logiku krokování tlaku. O této funkci více v příslušné sekci práce. Celé dopočítávání termodynamických veličin pak zařizuje funkce *dopocetNeznamych*.

Tato funkce obsahuje veškeré dopočty neznámých hodnot, respektive volá vhodné kombinace funkcí, které dopočítávají jednotlivé termodynamické veličiny v daném bodě. Vstupem funkce jsou dvě hodnoty a kombinace stavových veličin (případ). V nadcházející tabulce jsou uvedeny všechny případy zadání, pro jejichž řešení je funkce vybavena nástroji.

případ	kombinace stavových veličin	případ	kombinace stavových veličin
1	h-p	9	h-x
2	h-s	10	s-x
3	t-p	11	t-v
4	h-t	12	p-v
5	s-t	13	h-v
6	s-p	14	s-v
7	t-x	15	x-v
8	p-x		

Tabulka 7.2: Kombinace statových veličin jako vstup funkce dopocetNeznamych. Tyto překlady jsou relevantní skrze celý program.

Možností zadání je celkem 15. Funkce se skládá z definice výchozích hodnot a funkce Matlabu switch s parametrem případu zadání. Neboť velká část řešení zadání jsou si podobná, tak je zde uveden pouze obecný způsob řešení a pár příkladů.

Z tabulky funkcí XSteam-u (tabulka 6.2) vyplývá, že velmi důležitou veličinou je tlak. Dalšími veličinami, které se v tabulce hodnot často vyskytují, jsou teplota, entalpie a entropie. Z toho vyplývá, že až na výjimky stačí jedno ze zadání p-T, p-h nebo p-s. Obecný postup je pak následující. Ze zadaných veličin, pokud mezi nimi není, je spočítán tlak. Pak jsou dopočítané ostatní veličiny.

Zadání h-s

Jako příklad je zde uvedeno zadání h-s (číslo 2). Ze vstupu funkce (ze zadání) je k dispozici entalpie a entropie. Prvním krokem je tedy dopočítat tlak. K tomu je využita funkce XSteam(p_hs). V tuto chvíli jsou známy 3 termodynamické veličiny v požadovaném bodě a jedou z nich je tlak. Tedy dle tabulky funkcí XSteam-u 6.2 je možné k dalším výpočtům použít několik ověřených funkcí. Těmi jsou funkce t_hs, v_ph, u_ph a x_ph. Tedy je možné tyto funkce knihovny XSteam zavolat s jejich příslušnými vstupními parametry a tím dopočítat ostatní termodynamické veličiny v daném bodě. Následně jsou tyto hodnoty přiřazené k výstupům funkce dopocetNeznamych. V kódu Matlabu pak výpočet neznámých termodynamických veličin při zadání h-s vypadá následovně:

```
switch pripad
    case 2
        h = in1;
        s = in2;
        p = XSteam('p_hs', h, s);
        v = XSteam('v_ph', p, h);
        t = XSteam('t_hs', h, s);
        x = XSteam('x_ph', p, h);
        u = XSteam('u_ph', p, h);
```

Problém nastává ve chvíli, kdy mezi zadanými hodnotami není tlak a v knihovně XSteam není funkce, která by umožňovala tlak ze zadaných veličin spočítat. V knihovně Xsteam jsou pro výpočet tlaku dostupné pouze funkce psat_T, p_hs a p_hrho. První z trojice je omezená na velmi specifickou oblast stavů vodní páry, a tedy je pro řešení obecného zadání nevhodná. Poslední z trojice funkcí pro výpočet tlaku také není vhodná, a to ze dvou důvodů. Zaprvé v poznámce této funkce je v dokumentu [12] uvedeno, že pracovní médium voda a vodní pára, je v oblasti vody téměř nestlačitelná a funkce počítající tlak z hustoty je proto velmi nepřesná. Zadruhé pak hustota není známa, takže i pro oblasti stavů, ve kterých je funkce p_hrho dostatečně přesná je pro účely této práce nepoužitelná. Zbývá tedy pouze funkce p_hs.

Další komplikace nastává v případě, kdy v zadání program nemá entalpii i entropii. V takovém případě neznáme vstupní hodnoty parametrů této funkce a není možné ji přímo použít. Pro tyto případy je vytvořena sada funkcí. Tyto funkce pracují s knihovnou XSteam a postupují iteračně. Tyto funkce jsou popsány v nadcházejících sekcích této práce.

Zadání t-v

Druhým typovým příkladem funkce dopočtu hodnot termodynamických veličin je případ t-v (číslo 11). Na tomto příkladu je předvedeno dvojí. Zaprvé způsob volání funkce dopočtu tlaku a zadruhé vlastnost některých dopočtových funkcí. Touto vlastností je rozdělení výpočtu na případy (oblasti) a dle těchto případů pak vybrat vhodnější výpočet ostatních termodynamických veličin:

```
    case 11 %t-v
        t = in1;
        v = in2;
        [p, prip] = dopocet_p_tv(t, v);
        if prip == 1
            h = XSteam('h_pT', p, t);
            s = XSteam('s_pT', p, t);
            x = XSteam('x_ph', p, h);
        elseif prip == 2
            h = dopocet_h_pv(p, v);
            s = XSteam('s_ph', p, h);
```

```
    x = XSteam('x_ph', p, h);
end
u = XSteam('u_ph', p, h);
```

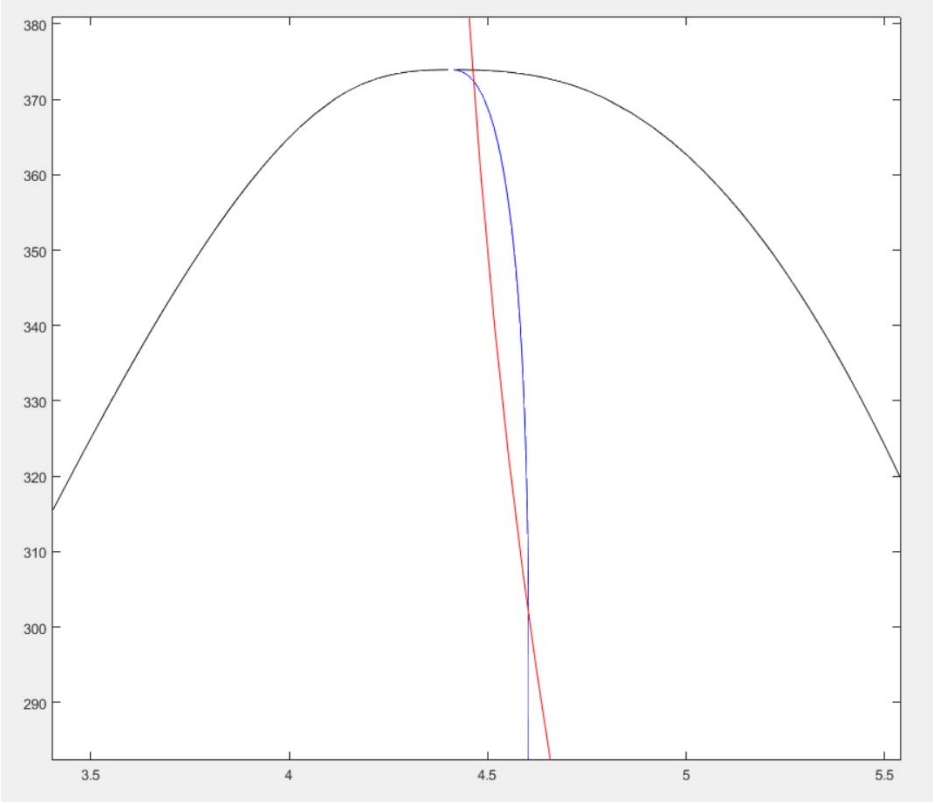
Funkce dopocet_p_tv má dvě návratové hodnoty. První návratovou hodnotou je hodnota tlaku, která je cílem výpočtu. Druhou návratovou hodnotou je pak dříve zmíněné rozdělení na oblasti. Při tomto konkrétním zadání (t-v) je výpočet rozdělen na případy, kdy tlak prochází oblastí mokré páry (případ 2) a nebo kdy tlak neprochází oblastí mokré páry (případ 1). Toto rozdělení bylo provedeno z důvodů schopnosti jednoznačně spočítat hodnoty požadovaných termodynamických veličin. Totiž v oblasti mokré páry isotermy a isobary splývají. V takovém případě nelze ze zadání teplota-tlak jednoznačně určit ostatní termodynamické veličiny.

K rozdělení pak došlo z důvodů přesnosti a rychlosti výpočtu. V případě, že hodnotu lze zaručeně spočítat pomocí funkcí XSteam, je tak spočítána. V případě, že není jednoznačné zadání, je použita iterační funkce dopocet_h_pv. Tím je dopočítán další parametr a zbylé veličiny je možné dopočítat přímo pomocí funkcí knihovny XSteam. Je totiž předpokládáno, že výpočet pomocí funkce knihovny XSteam je přesnější a rychlejší než výpočet pomocí iterační funkce založené na funkcích knihovny XSteam.

Vhodnějším řešením by bylo prvně se pokusit spočítat hodnoty přímo pomocí knihovny XSteam a iterační funkci dopocet_h_pv použít pouze v případě nezdaru. Tato implementace by byla vhodnější z výpočetního hlediska. Ale knihovna XSteam v těsném okolí mezní křivky (x=0 a x=1) v některých případech vrací zjevně nesmyslné hodnoty. V případě tohoto řešení by bylo nutné hlídat nejen korektní výpočet, ale i absolutní hodnotu návratové veličiny. Algoritmus, který by prováděl tuto kontrolu, by byl výrazně složitější než v použitém rozdělení. Časová náročnost tvorby a testování tohoto algoritmu by také odpovídala jeho složitosti. Podobných problémů je skrze program více a bohužel řešení všech takových případů je mimo časové možnosti dané zadáním této práce.

Zadání h-x

Dalším příkladem výpočtu termodynamických parametrů se specifickým problémem je dopočet dle zadání h-x (číslo 9). Tento případ je specifický tím, že nemusí být jednoznačně řešitelný. Křivky suchosti kolem hodnoty $0,5$ na sobě v t-s diagramu mají "vlnu". Kombinace této křivky a zakřivení izoentalp umožňuje existenci dvou průsečíků. Problém je znázorněn na následujícím obrázku.



Obrázek 7.4: Grafické znázornění problému výpočtu při zadání h-x (číslo 9). Výřez t-s diagramu.

Funkce dopocet_p_hx začíná iterační proces na hodnotě tlaku $p = 0,00611658\,bar$ a krokuje směrem k hodnotě vyššího tlaku. Tím pádem, pokud nejsou body velmi blízko u sebe, nalezne vždy bod při nižší teplotě (tlaku). Pokud jsou body blízko u sebe tak je možné že iterační krok "překročí"oba dva body najednou. V takovém případě není možné rozhodnout, který z bodů byl nalezen. V takovém případě jsou ale body blízko u sebe a je otázkou, jestli je nepřesnost způsobená nalezením druhého bodu podstatná. Základní velikost iteračního kroku funkce dopocet_p_hx je 1 *bar*. Praktické provedení výpočtu ve funkci *dopocetNeznamych* je následující:

```
case 9 %h-x
    warning('Zadání h-x má 2 možné průsečíky, zatím neumím najít oba!!')
    h = in1;
    x = in2;
    p = dopocet_p_hx(h, x);
    s = XSteam('s_ph', p, h);
    v = XSteam('v_ph', p, h);
    t = XSteam('T_ph', p, h);
    u = XSteam('u_ph', p, h);
```

V příkladu kódu je zaznamenáno varování vypsané do konzole. Varování říká, že funkce zatím nemá implementovány možnosti pro řešení neurčitých případů. K implementaci těchto možností nedošlo z důvodu časového omezení práce. Jedním z možných řešení by bylo v rámci uživatelského rozhraní programu vytvořit takzvané vyskakovací okno, které by bylo zobrazeno v případě tohoto neurčitého zadání. Okno by obsahovalo všechny termodynamické parametry obou bodů a možnost výběru jedné ze sad hodnot pro další běh programu.

Stejný problém nastává při zadání s-x (číslo 10). Funkce *dopoceNeznamych* řeší toto zadání podobně, jako případ zadání h-x. Při zadání s-x také není vyřešené nelezení obou řešení a následné rozhodování.

Řešení problému by bylo podobné jako bylo naznačeno u řešení zadání h-x, a to vyskakovacím oknem s možností výběru.

Zadání s-x

V případě zadání s-x (číslo 10) došlo k několika fázím vývoje. Funkce nedopočítávala požadované hodnoty a končila v nekonečných smyčkách. V tuto chvíli funkce funguje tak jak má, ale pozůstatkem vývoje je rozdělení funkce na dvě. Těmito funkcemi jsou *dopocet_p_sx_advanced* a *dopocet_p_sx_pomoc_p*. Hlavní z těchto dvou funkcí je *dopocet_p_sx_advanced*. Podřazená funkce je v tuto chvíli volána pouze z funkce *dopocet_p_sx_advanced* a je možné ji do této funkce snadno vnořit a soubor *dopocet_p_sx_pomoc_p.m* následně smazat.

Nicméně v tomto stavu funkce funguje tak jak má a tedy toto zjednodušení není považováno za důležité. Tímto daná priorita vedla k odsunutí řešení na pozdější datum. Pak z důvodů časového omezení daného termíny této práce nebylo toto zjednodušení provedeno.

7.2.3 Dopočtové funkce

Většina funkcí vytvořených pro doplnění funkcí knihovny XSteam jsou si velmi podobné. Používají stejný princip k nalezení požadované termodynamické veličiny. Funkce využívají iterační metody založené na funkcích knihovny XSteam a to tak, že hledaná termodynamická veličina je vstupní hodnotou výpočetní funkce, jedna ze známých (zadaných) veličin je druhou vstupní hodnotou a druhá ze zadaných veličin je hodnotou výstupní. Postup je pak takový, že funkce krokuje hledanou veličinu tak dlouho, dokud výstupní hodnota z funkce knihovny XSteam není v blízkosti hodnoty zadané.

Funkce dopocet_h_pv

Praktické řešení je ukázáno na příkladu funkce *dopocet_h_pv*. Funkce je použita pro dopočet hodnoty entalpie ze zadání t-v (číslo 11) za případu, že tlak nalezený funkcí *dopocet_p_tv* prochází oblastí mokré páry.

V případě zadání t-v známe teplotu a objem. První dopočítanou hodnotou je tlak. Pro další výpočty je zapotřebí znát ještě jednu další termodynamickou veličinu. Z tabulky funkcí XSteam-u 6.2 vyplývá, že vhodnou veličinou je entalpie. Tedy byla vytvořena funkce *dopocet_h_pv*.

Uvnitř této funkce je po definici konstant a proměnných výchozího stavu while cyklus. Podmínkou ukončení tohoto cyklu je odchylka vypočítané veličiny objemu od její známé (zadané) hodnoty. Druhou podmínkou ukončení cyklu je velikost kroku. V případě, že by se funkce dostala do neočekávaného stavu a cyklus by nebyl ukončen úspěšným nalezením hodnoty, je cyklus ukončen na podmínku velikosti kroku. Tato podmínka ukončení je relevantní například v případě zadání velmi přesné hodnoty objemu. V takovém případě se může stát, že hodnoty vypočítané pomocí funkce knihovny XSteam má menší přesnost, než je zadaná hodnota, respektive nastavené odchylka. V případě, že se nepodaří úspěšně nalézt hodnotu v oblasti, pro kterou jsou funkce knihovny XSteam definovány, je navrácena hodnota *NaN*. Pro takový případ je do funkcí implementována pojistka zajišťující zmenšení kroku. Tím je zaručeno ukončení cyklu na podmínku velikosti kroku. V takovém případě je z výpočetní funkce navrácena hodnota *NaN*, která je programem dále zpracovávána.

Uvnitř while cyklu je proveden rozdíl zadané hodnoty objemu a hodnoty vypočítané pomocí již známé hodnoty tlaku a hledané (krokované) entalpie. Do funkce *dopocet_pomoc_p* je pak zadáno krokování veličiny h a příslušné pomocné proměnné. O funkci *dopocet_pomoc_p* více v následující sekci této práce. V kódu pak while cyklus funkce *dopocet_h_pv* vypadá následovně:

```
while (dist > citlivost && krok > citlivost)
    dist = v - XSteam('v_ph', p, h);
    [h,dist,smer,krok,last] = dopocet_pomoc_p(h,dist,citlivost,smer,krok,last,2);
end
```

Funkce dopocet_pomoc_p

Tato funkce má poněkud zavádějící název pomoc_p. Název naznačuje, že se jedná o pomocnou funkci pro výpočet tlaku. Zde ale označení písmenem p představuje "proměnné". Název funkce je "dopočet pomocné proměnné". Funkce dopocet_pomoc_p tedy krokuje libovolné veličiny (proměnné). Její definice vypadá následovně:

```
function [p, dist, smer, krok, last] = dopocet_pomoc_p(p,...
                                          dist, citlivost, smer, krok, last, pripad)
```

Funkce má několik vstupů a několik výstupů. První vstupní hodnotou funkce je *p* jako proměnná. Jedná se o krokovanou veličinu.

Druhou vstupní hodnotou je *dist*, z anglického distance (vzdálenost). To je hodnota rozdílu mezi zadanou hodnotou porovnávané veličiny a hodnotou této veličiny vypočítané pomocí funkcí knihovny XSteam. Tento rozdíl není v absolutní hodnotě, tj. může být záporný, a je využit pro rozhodnutí směru kroku proměnné.

Třetí vstupní proměnnou je *citlivost*. Ta určuje povolenou velikost absolutní hodnoty proměnné dist. Tedy pokud je absolutní hodnota dist menší než citlivost, tak je krokování považováno za dokončené a v tomto cyklu nadřazené výpočetní funkce funkci *dopocet_pomoc_p* již není krokováno. Pokud by byl krok proveden, tak je možné, že by se vypočtená hodnota porovnávané veličiny od hodnoty zadané vzdálila.

Čtvrtou vstupní proměnnou je *smer*. Název proměnné vychází z českého slova směr. Jedná se o proměnnou určující směr posledního kroku. Proměnná směr je nepřímo využita pro detekci změny znaménka proměnné *dist*. Pokud dojde ke změně směru kroku, tak je krok zmenšen. Bez této proměnné by funkce *dopocet_pomoce_p* neměla možnost poznat, jestli je vhodné zmenšit krok.

V minulém odstavci byl zmíněn krok. Ten je i další, v pořadí již pátou, vstupní proměnnou funkce *dopocet_pomoc_p*. Proměnná *krok* slouží k takovému účelu, k jakému napovídá její název. Jedná se o proměnnou obsahující aktuální velikost kroku krokované termodynamické veličiny.

Název šesté vstupní proměnné je pak převzat z anglického překladu slova poslední, *last*. Jedná se o dříve zmíněnou pojistku přerušení cyklu. Pokud je aktuální hodnota stejná jako poslední vypočítaná hodnota, tedy nedošlo ke kroku nebo došlo k chybě výpočtu a obě hodnoty jsou rovny *NaN*. Tato proměnná zajišťuje výrazné zmenšení kroku. To velmi rychle vede k ukončení cyklu nadřazené funkce na podmínku velikosti kroku.

Poslední, sedmou proměnnou je *pripad*. Jedná se o velmi důležitou proměnnou. Určuje o jaký typ krokování se jedná. Totiž pokud je proměnná *dist* kladná, tak některé veličiny vyžadují krok směrem k vyšším hodnotám krokované veličiny a některé vyžadují krok směrem k hodnotám nižším. Tato veličina funkci *dopocet_pomoc_p* určuje, kterým směrem má pro která znaménka proměnné *dist* krok provést. Logika směru kroku a přiřazení k hodnotám případu je následující:

```
% pripad = 1 -> dist > 0; p - krok
% pripad = 2 -> dist > 0; p + krok
```

Je zřejmé, že některé ze vstupních proměnných se během krokování mohou změnit. Aby bylo možné tyto změněné proměnné mít opět jako vstupní, tak je nutné tyto proměnné vracet nadřazené funkci. První z návratových hodnot je samozřejmě krokovaná proměnná.

Druhou výstupní proměnnou je *dist*. Do té je pro účely koncové podmínky nadřazené funkce, ukládána absolutní hodnota *dist*. Jinak by buďto bylo zapotřebí absolutní hodnotu dist provést ve všech nadřazených funkcích nebo by cyklus krokování byl předčasně ukončen na první záporné hodnotě rozdílu hodnoty zadané veličiny a hodnoty spočítané veličiny.

Třetí výstupní proměnnou je směr (*smer*). Z vysvětlení vstupní hodnoty této proměnné je zřejmé proč je i mezi výstupními proměnnými. Pro rekapitulaci je pod touto proměnnou uložený směr kroku který je žádoucí mít pro rozhodnutí směru a velikosti příštího kroku.

Krok je čtvrtou návratovou proměnnou. U této veličiny jako vstupní veličiny bylo vysvětleno že se jedná o velikost kroku. Ta je během cyklování měněna a tedy je nutné ji pro další krokování znát.

Poslední návratovou veličinou je poslední (*last*). Z vysvětlení této proměnné jako vstupní veličiny je zřejmé, že tuto proměnnou je zapotřebí ukládat pro další kolo cyklu.

Jeden případ krokování je pak implementován takto:

```
switch pripad
    case 1
        if abs(dist)<citlivost
            %continue
        elseif dist > 0
            % chci p-
            smer = 1;
            p = p - krok;
        elseif dist < 0
            % chci p+
            smer = 0;
            p = p + krok;
        elseif dist == 0
            % continue
        else %musí být nan
            if smer == 0
                p = p + .5 * krok;
            elseif smer == 1
                p = p - .5 * krok;
            end
            krok = krok * .1;
            dist = 100;
        end
```

Druhý případ (case 2) je analogií prvního případu (case 1) pouze s otočenými znaménky u krokování, tj. v případě $dist > 0$ je provedeno $p + krok$.

Vnitřní energie

Funkce pro výpočty pomocí zadání vnitřní energie jsou velmi podobné funkcím pro výpočty při zadání objemu. Problémem je pak provedení funkce pro výpočet při zadání objemu a vnitřní energie. Při takovém zadání není čeho se chytit a je velmi náročné vytvořit takovou výpočetní funkci. Jediná cesta, která mne napadá je vytvořit vnořené krokování. Tedy vytvořit funkci, která by pomocí funkcí knihovny XSteam krokovala ne jen jednu vstupní veličinu, ale obě dvě. Taková funkce by byla buďto velmi nepřesná, nebo velmi pomalá. Dokonce je možné, že by funkce byla jak velmi pomalá, tak velmi nepřesná.

Po konzultaci s vedoucím práce bylo rozhodnuto, že zadání pomocí vnitřní energie nejsou často využívána a tedy není nutné je do programu implementovat. Stačí, když program bude vnitřní energii počítat z ostatních zadání a zobrazovat ji společně s ostatními termodynamickými veličinami. Dopočet vnitřní energie, pokud známe ostatní termodynamické veličiny, je pomocí funkcí knihovny XSteam velmi snadný. Máme k dispozici funkce u_pT, u_ph a u_ps. Tedy stačí znát jednu z těchto kombinací veličin. Tento výpočet je pak proveden v rámci funkce *dopocetNeznamych*.

Vývoj funkcí pro výpočty pomocí zadání vnitřní energie byl na základě této konzultace zastaven a funkce nebyly do programu implementovány. Již plně hotové, i jen částečně hotové, funkce pro tento typ výpočtů jsou pro případ budoucího vývoje programu přiložené k ostatním funkcím ve složce dopocty_hodnot v hlavní složce programu.

7.2.4 Vykreslení děje

Samotné vykreslení dějů je snadné. Stačí vzít osy diagramu a zavolat funkci Matlab-u *plot*. Osy se nacházejí v okně uživatelského rozhraní a tím této funkci přibývá jeden parametr. Tím je právě do kterých os funkce vykresluje. Volání funkce plot pro vykreslení děje pak vypadá následovně

```
plot(fig, x, y, "LineWidth", linewidth, "Color", barva);
```

Postupně definujeme právě zmíněný cíl vykreslení fig (osy), hodnoty na x-ové a y-ové souřadnici (2 listy) a grafické modifikace základního nastavení. Těmi jsou tloušťka čáry (linewidth) a barva.

Číselné vyjádření děje je provedeno právě seznamy (listy) hodnot. Tedy například pro vykreslení děje v t-s diagramu jsou vypočítané dvojice hodnot t a s ležící na křivce děje. Počet bodů musí být přiměřený s rozumným rozdělením. Pro účely této práce bylo rozhodnuto že 50 hodnot rozdělených lineárně dle teploty je dostačující. Počet bodů i jejich rozdělení se ale v závislosti možností vhodného výpočtu u některých funkcí liší.

Výpočet bodů

Výpočet bodů pro vykreslení křivky děje je proveden pomocí funkce *switch_dej*. Vstupními proměnnými jsou celá aplikace app (její proměnné), případ zadání a typ děje. Případ zadání je číslo 1 až 15 dle tabulky 7.2 a typ děje je string z dropdown menu dějů (izotermický, izobarický, ...). Pomocí těchto proměnných funkce rozhoduje který výpočet provést. Výpočet pak vychází z hodnot jednotlivých termodynamických veličin zadaného bodu (ps) a zadaného bodu (ks).

Výstupy této funkce jsou pak listy hodnot pro zobrazení děje v h-s a t-s diagramu, a hodnoty termodynamických veličin v koncovém bodě (ks). Před voláním funkce *switch_dej* je v koncovém stavu známa pouze jedna, uživatelem zadaná veličina. Ostatní jsou nastavené na výchozí hodnotu -1. Teprve v rámci běhu funkce jsou spočítány hodnoty termodynamických veličin koncového stavu. Proto jsou i mezi výstupy této funkce.

Volání funkce *switch_dej* je pak díly velkému množství vstupních i výstupních proměnný velmi dlouhé. Proto je následující příklad rozdělen do dvou řádků.

```
function [outH_hs, outS_hs, outT_ts, outS_ts, t2, p2, v2, h2, u2, s2, x2] = ...
 swtich_dej(app, pripad, dej, t, p, v, h, u, s, x, t2, p2, v2, h2, u2, s2, x2)
```

Vzhledem k povaze proměnných by bylo vhodnější nezadávat parametry ps a ks po jednom, ale využít struktury. Takovou strukturou by mohla být takzvaná *class*. Bylo by možné použít class *bod_class* (k této class více v příslušné kapitole). Ale z historických důvodů vývoje funkce a časového omezení nebylo toto zjednodušení implementováno.

První částí samotné funkce je mnohařádková funkce switch vnořená do jiné switch. Cílem tohoto konstruktu je pomocí proměnné *dej* (nadřazená switch) a pomocí proměnné *pripad* (několik vnořených switch) přiřadit vstupní hodnoty konkrétních proměnných (termodynamických veličin) k obecným proměnným *in1* a *in2*. Nadřazená switch (děj) dělí případy zadání dle děje vybraného uživatelem. Možné případy jsou p, t, v, s, adiabata a h. Pak ve vnořené switch jsou již jen případy obsahující veličinu definovanou vybraným dějem. Například v případě isobarického děje je proměnná *dej* rovna hodnotě 1. Následuje vnořená switch s možnostmi h-p, t-p, s-p, p-x a p-v. Například v případě že uživatel zadal v ks hodnotu entalpie. Pak je ve vnořené switch vybrán případ h-p (číslo 1).

Termodynamické veličiny jsou převáděny na jednotky, se kterými knihovna XSteam umí pracovat, při jejich načtení. Tedy v okamžiku jejich předání této funkci jsou již převedeny. Tedy je možné přiřadit i jednotky *unit1* a *unit2*. Jednotky jsou zapotřebí pro výpočetní funkce využité ve funkci *switch_dej*. Příklad pár řádků tohoto switch konstruktu je zobrazen níže.

```
switch dej
    case 1 %p
```

```
        switch pripad
            case 1 %hp
                in1 = h2;
                in2 = p;
                unit1 = "kJ/kg";
                unit2 = "bar";
            case 3 %tp
... %pokračování dalších case pripad
        end
    case 2 %t
        switch pripad
     ...% pokračování další case dej
```

Po této velké switch následuje dopočet hodnot ks a další mnohařádkový switch. Úkolem tohoto druhého switch je rozhodnout, který výpočet bodů čáry děje provést:

```
switch dej
    case 1 %p - izobarický děj
        [outH_hs, outS_hs, outT_ts, outS_ts] = isobaricky_fce(p, s, h, s2, h2);
    case 2 %t - izotermický děj
        ... % další case dej
```

Případy zadání děje p, t, v, s a h jsou si vzájemně velmi podobné, respektive postup výpočtu bodů děje odpovídajícího zadání. Náročnější proces nastává při zadání adiabatického děje. V takovém případě je ještě vypočítán koncový stav. Program totiž za výchozí koncový bod považuje adiabatický děj s účinností 1, neboli izoentropický děj. V rámci výpočtu adiabatického děje je pak pomocí tohoto výchozího bodu a uživatelem zadané účinnosti spočítán platný koncový stav.

Adiabatický děj

Řešení případu zadání adiabatického děje je rozdělené dle případu zadání koncové veličiny. Tedy výpočet se liší pro zadání t, p, v, h a x. Výpočty probíhají iterační metodou podobnou metodě zmíněné v sekci Izochory. Každé ze zadání je pak dále rozdělené na případ komprese a expanze.

Řešení tohoto problému je (na počet řádků) velmi dlouhé. Jednotlivé sekce kódu jsou si ale vzájemně podobné, a tak je možné zde uvést pouze jeden případ zadání. Pro účely příkladu je uvažováno, že uživatel zadal adiabatickou expanzi s koncovým bodem entalpie. Pak dle rovnice:

$$h_2 = h_1 - \eta(h_1 - h_2') \tag{7.1}$$

Zde h_1 představuje entalpii počátečního stavu, h_2' představuje entalpii koncového stavu při účinnosti 1 a η představuje účinnost (zlomek). Tento výpočet je uvnitř while cyklu který krokuje h_2' tak, aby h_2 bylo rovno hodnotě entalpie koncového stavu. Po nalezení tohoto pomocného bodu h_2' je pomocí s_1 dopočítán tlak. Pomocný bod předpokládá adiabatický děj o účinnosti 1. Pak je z pomocného bodu do reálného koncového proveden posuv po izobaře. Tedy hledaný koncový bod je nalezen jako kombinace zadané hodnoty entalpie a tlaku pomocného bodu. Následně jsou mezi počátečním a právě spočítaným koncovým bodem spočítány body pro vykreslení čáry adiabatického děje. Provedení řešení tohoto případu zadání pak vypadá následovně:

```
switch pripad
    case 2 % h-s
        if h2 > h %pripad expanze
            h2_ref = h2;
            dist = 1000;
```

```
            h_krok = 100;
            while (abs(dist) > .001 && h_krok > .000001)
                h2 = h  - ucinnost * (h - h2_);
                dist = h2_ref - h2;
                if abs(dist) > .001
                    if dist < 0
                        h2_ = h2_ - h_krok;
                    elseif dist > 0
                        h_krok = h_krok / 2;
                        h2_ = h2_ + h_krok;
                    elseif dist == 0
                        % do nothing
                    end
                end
            end
            [t2, ~, v2, s2, x2, ~, u2] = dopocetNeznamych(h2, p2_, 1);
            [outH_hs, outS_hs, outT_ts, outS_ts] = adiabaticky_fce(ucinnost, h, t, s, h2_, s2);
            h2 = h_ref;
        elseif h > h2 %pripad expanze
            ... % expanze a pak další případy (case)
```

Výpočet bodů na křivce zadaného adiabatického děje pak probíhá pomocí funkce *adiabaticky_fce* následovně. Po inicializaci funkce následuje výběr výpočtu komprese nebo expanze. Výpočet je pak pro každý z bodů na izočáře proveden pomocí *for* cyklu. Uvnitř cyklu je opakován výpočet s využitím pomocného bodu. V této funkci je ale již z rozdělení bodů na křivce známá hodnota entalpie a tedy není nutné využít iterační metody. Po nalezení koncového bodu jsou dopočítány ostatní veličiny potřebné pro vykreslení h-s a t-s diagramů. Níže je uvedený příklad kódu výpočtu bodů adiabatické expanze:

```
if h1 > h_end % expanze
    %spocitej krok a opakuj
    %i = 2;
    for i = 2:bodu
        %carkovany bod
        s2_ = s1;
        h2_ = list_h(i);
        p2_ = XSteam('p_hs',h2_,s2_);
        %necarkovany bod
        h2 = h1 - ucinnost * (h1 - h2_);
        s2 = XSteam('s_ph',p2_,h2);
        %zapis vysledky
        out_h(i) = h2;
        out_s(i) = s2;
        out_T(i) = XSteam('T_hs',h2,s2);
    end
elseif ... % komprese
```

První hodnota na vykreslené adiabatě je počáteční stav. Ten je již zadaný a není nutné ho počítat. Proto *for* cyklus výpočtu začíná druhou hodnotou v listu.

Funkce plotDeje

Funkce *plotDeje* má jeden úkol. Tím je ze vstupních listů hodnot vykreslit děj. Této funkci ale může být nadřazených několik jiných. Nadřazená funkce ovlivňuje položky a způsob vykreslení děje. Funkce

plotDeje je vybavená možnostmi pro všechny tyto případy. Jedná se o komplikovanou funkci a nebude zde podrobně popsána.

Funkce *plotDeje* provádí několik operací. Jedná se o načtení typu děje, přiřazení popisku (čísla) uvedeného u čáry děje, načtení typu diagramu (h-s, t-s), dle děje přiřazení barvy čáry děje, kontrola platnosti vstupních listů hodnot, nastavení různých grafických detailů jako je tloušťka čáry a samotné vykreslení čáry děje. Každá z operací je složená z několika jednoduchých, srozumitelných příkazů. Komplikovanou částí funkce je logika rozhodování, které operace je v daném případě jejího volání žádoucí provést.

7.2.5 Další funkce programu

Program obsahuje mnoho pomocných funkcí jako jsou funkce obsluhující kontrolu zadání a oblasti ve které program počítá termodynamické veličiny. Některé z těchto funkcí pracují na pozadí programu a uživatel o nich při vhodném použití programu neví. Takové funkce zde nebudou popsány.

Dále jsou v této práci popsány funkce které mají přímý dopad na uživatele. Takovými funkcemi jsou paměť programu a s ní spojené kopírování termodynamických veličin daného stavu do schránky (ctrl+C) a funkce pro automatické zobrazování (posouvání) diagramů. Tyto funkce využívají následující struktury.

Struktura bod

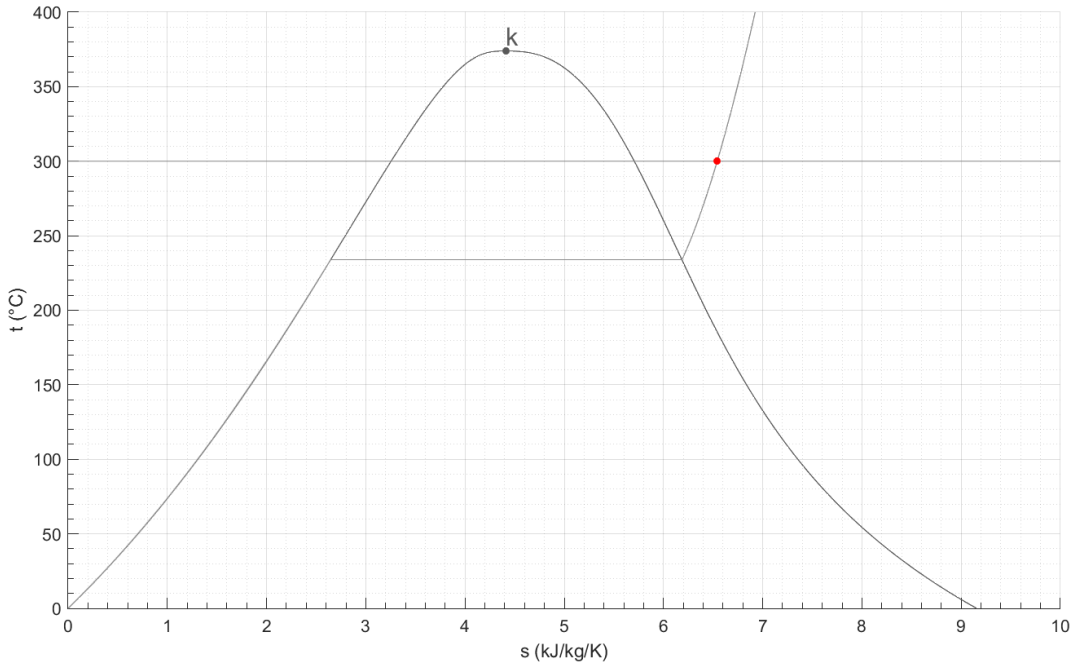
Jedná se o strukturu, v jazyce Matlabu o takzvaný *class*, pomocí které je v programu definovaný jeden termodynamický stav, tj. stavy počáteční a koncový. Struktura obsahuje *properties* (vlastnosti). To jsou jednotlivé proměnné této struktury. Konkrétně jsou to termodynamické veličiny t, p, v, h, s, x a u. Dále struktura obsahuje zadání pomocí kterého byl bod spočítán. Zadání bodu má program definované dvojí. Počáteční stav a koncový stav. Počáteční stav je v programu definován pomocí hodnot dvou termodynamických veličin a koncový stav je definovaný pomocí děje a jedné termodynamické veličiny. Další vlastností struktury stavu je další struktura. V té jsou uloženy jednotky uložených termodynamických veličin. Dále struktura obsahuje několik pomocných funkcí jako jsou funkce pro načtení stavů a jednotek.

Struktura bod pak obsahuje odkaz na objekt v diagramu. Respektive na bod jako značku vyjadřující uložený termodynamický stav. Pro případ adiabatického děje struktura obsahuje ještě odkazy na pomocné grafické objekty v diagramu. Tedy například odkaz na pomocný (čárkovaný) bod. Z technických důvodů, do kterých zde nebudeme zabředat, struktura ještě obsahuje odkaz na popisek děje.

Struktura, jako class Matlabu, umožňuje definici funkcí operujících s daným kusem objektu této struktury. Tedy například je možné definovat funkci:

```
function [t, p, v, h, u, s, x] = getVals(obj)
    t = obj.t;
    p = obj.p;
    v = obj.v;
    h = obj.h;
    u = obj.u;
    s = obj.s;
    x = obj.x;
end
```

která z dané struktury (objektu) přečte termodynamické veličiny a po jedné je vrátí nadřazené (volající) funkci v tomto daném pořadí. Tato a ostatní funkce této struktury jsou definovány jako *methods* (metody) tohoto objektu. Mezi podstatné funkce zde patří funkce obsluhující pomocné čáry v diagramu. Především funkce *vyrob_pomocne*, která k danému bodu vytvoří pomocí zadání (z proměnných struktury) pomocné čáry. Těmi jsou nevýrazné čáry v diagramu na jejichž průsečíku leží daný bod. Tedy například při definici pomocí tlaku a teploty tato funkce vytvoří pomocnou izobaru a izotermu. Tyto izočáry odpovídají uživatelem zadaným hodnotám.



Obrázek 7.5: Ukázka pomocných čar v t-s diagramu pro zadání $t = 300\,°C$ a $p = 30\,bar$.

Struktura děj

Struktura děje obsluhuje uživatelem zadaný termodynamický děj. Struktura obsahuje proměnné pro typ děje, dva body (bod class) s možným třetím pro pomocný bod adiabatického děje, struktury minimálních a maximálních hodnot a ukazatel na křivku děje v diagramu.

Typ děje zaznamenává jaký děj uživatel vybral, a tedy jaký děj je v počítán. Struktury maximálních a minimálních hodnot jsou vytvořené za účelem kontroly hodnoty uživatelem daného zadání. Toto zadání musí být v mezích, ve kterých knihovna XSteam, a tedy i program, může pracovat. Například není možné zadat teplotu 2000 $°C$. Část omezení výpočetní oblasti vychází ze standardu IAPWS-IF97, část z implementace tohoto standardu knihovnou XSteam a část z uměle vytvořeného omezení programu, jejichž cílem je umožnit jednoznačnou definici hranice výpočetní oblasti pomocí jedné funkce.

Samotná kontrola mezí pak probíhá pro různé stavy programu jinak, respektive jsou prováděny jiné výpočetní úkony. Po zadání první termodynamické veličiny počátečního stavu je provedeno porovnání mezí výpočetní oblasti a této zadané hodnoty. Pokud zadaná hodnota leží mimo meze výpočetní oblasti, tak v programu vyskočí chybová hláška obsahující povolené meze dané termodynamické veličiny. Pokud je zadaná hodnota v mezích výpočtové oblasti tak tuto hodnotu program přijme a přepočítá meze jednotlivých termodynamických veličin tak, aby ostatní termodynamické veličiny měly na svých izočárách průsečík s izočárou již zadané termodynamické veličiny. Například zadání entalpie omezí možné hodnoty teploty. Po zadání druhé termodynamické veličiny počátečního stavu opět dojde ke kontrole mezí a případně také vyskočí okno s informací o platných mezích. Pokud je druhá hodnota počátečního stavu programem přijata, tak je proveden další výpočet mezí. Tentokrát pomocí počátečního stavu a zvoleného děje. Tento přepočet je proveden při každé změně děje. Tím je zajištěno, že program ze zadaných hodnot vždy může spočítat parametry koncového stavu.

Tedy program při každé změně zadání přepočítá platné meze každé termodynamické veličiny, kterou je možné zadat. Funkce, která toto počítání zařizuje, je definována jako metoda struktury děje. V rámci této funkce jsou pomocí *switch* definována všechna možná zadání, respektive výpočty mezí prostřednictvím zadané hodnoty. V rámci struktury jsou definovány i další funkce určené k její obsluze. Ty ale uživatele přímo neovlivňují a nebudou zde podrobně popsány. Příkladem takové funkce je inicializace (před-vyplnění) struktury.

Paměť

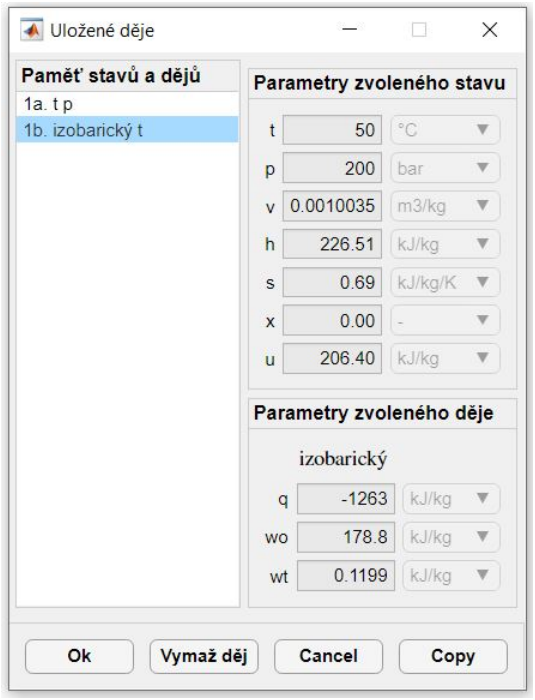
Program pro splnění všech požadavků požadavků musí obsahovat paměť. Pro plnou funkčnost programu byla paměť rozdělena na dvě části. Pro samotné fungování programu, ve smyslu schopnosti programu ze zadaných hodnot spočítat koncový stav, je důležitá takzvaná paměť dějů. Pro rozšiřující funkce programu, jako je změna diagramu nebo jeho editace, je implementována takzvaná paměť bodů.

Paměť dějů je implementací struktury *dej_class*. Jedná se o paměť, jejíž cílem je plně obsluhovat jeden děj. Přesněji jsou v ní uložená veškerá data daného děje. Tato paměť tedy obsahuje vše od uživatelem zadaných veličin a jejich pořadí až po dopočítané meze nebo ukazatel na popisek čáry děje v diagramu. Paměť je vyplňována a měněna v závislosti na interakci uživatele s uživatelským rozhraním. Tedy pokud uživatel zadá hodnoty počátečního stavu a stiskne tlačítko *Hledej*, tak program do paměti dějů uloží do proměnné *bod1* zadané a vypočítané parametry. Pak po výběru děje, vyplnění hodnoty koncového stavu a stisku tlačítka *Vypočti* jsou uloženy ostatní položky.

Paměť bodů je pak list struktur *bod_class*. Hodnoty dějů jsou v této paměti uložené po dvojicích, každý děj má uložený počáteční stav (bod) a koncový stav. Každý bod, vyjma prvního počátečního stavu, je tedy duplikován. Tato vlastnost je částí řešení problematiky editace dějů v paměti. Body jsou do této paměti uloženy pouze po stisku tlačítka *Ulož*. Paměť slouží jako "záloha"zadaných bodů. Například při změně diagramu z t-s na h-s dojde k načtení prázdného h-s diagramu. Veškeré vykreslené body a děje jsou tím ztraceny. Tím by uživatel přišel o veškerou svou dosavadní práci v programu. S pomocí této paměti pak program veškeré doposud zadané (uložené) děje rekonstruuje v příslušném diagramu. Paměť je možné editovat v okně paměti. To je možné otevřít stiskem tlačítka *Paměť*. Toto okno je vytvořené pomocí alternativy k prostředí App designer Matlabu tak, že je okno a každý jeho prvek definován ručně pomocí příkazů. Příkladem takového příkazu je *uipanel*() nebo *uibutton*();.

Okno paměti umožňuje procházet dříve zadané stavy a děje a umožňuje editaci paměti bodů ve smyslu odstranění děje. V okně paměti je na levé straně seznam stavů. V páru jsou značené číslem a písmenem. Číslo značí děj a písmena a, b znační počáteční, respektive koncový stav daného děje. Pravá část okna je pak rozdělena na dvě části. V horní části jsou zobrazeny termodynamické veličiny v daném bodě. V dolní části pak parametry zvoleného děje q (sdělené teplo), wo (objemová práce) a wt (tlaková práce). Přepnutím stavu v rámci jednoho děje se změní zobrazené parametry v pravé horní části okna. Pravá dolní část okna se změnou stavu v rámci děje nezmění. Přepnutím děje se změní obě části zobrazovaných parametrů.

V dolní části okna jsou tlačítka *Ok*, *Vymaž děj*, *Cancel* a *Copy*. Tlačítko *Ok* Potvrdí výběr děje a do kolonek počátečního stavu v hlavním okně programu načte parametry děje vybraného v okně paměti. Tlačítko *Vymaž děj* z paměti odstraní dvojici bodů odpovídající zvolenému ději a překreslí diagram tak, aby v něm nebyl zobrazený právě odstraněný děj. Tlačítko *Cancel* zavře okno paměti aniž by s ní nebo s hlavním oknem provádělo pro uživatele a termodynamické děje něco podstatného (změní se nastavení tlačítek blokující otevření druhého okna paměti). Nakonec pak tlačítko *Copy* do schránky kopíruje ("Ctrl+C") parametry počátečního a koncového stavu vybraného děje a jeho parametry. Tedy je po stisku tlačítka copy možné v Excelu, nebo libovolném textovém editoru stisknout známou klávesovou kombinaci "Ctrl+V"a parametry tím do daného prostředí vložit. Grafické provedení okna paměti je následující:



Obrázek 7.6: Grafické provedení okna paměti programu

Původní záměr implementace paměti byl takový, že v paměti programu budou uložené jednotky které uživatel zvolil. To by pak umožnilo načtení hodnot tak, jak je uživatel uložil. Technické provedení tohoto řešení, respektive převodů jednotek v několika okamžicích běhu programu, se ale ukázalo náročnější než bylo původně předpokládáno. To vedlo ke zvýšení časové náročnosti implementace této funkce programu. Tuto časovou ztrátu nebylo možné si v rámci této práce dovolit. Proto je v této implementaci programu problém řešen tak, že se do paměti vždy ukládají hodnoty v jednotkách knihovny XSteam. To vede k tomu, že hodnoty jsou vždy načteny v jednotkách knihovny XSteam. Uživateli je pak umožněno změnit načtené jednotky dle jeho uvážení. Bylo by vhodné v rámci případného dalšího vývoje programu tuto funkci doplnit.

7.3 Příklad práce s programem

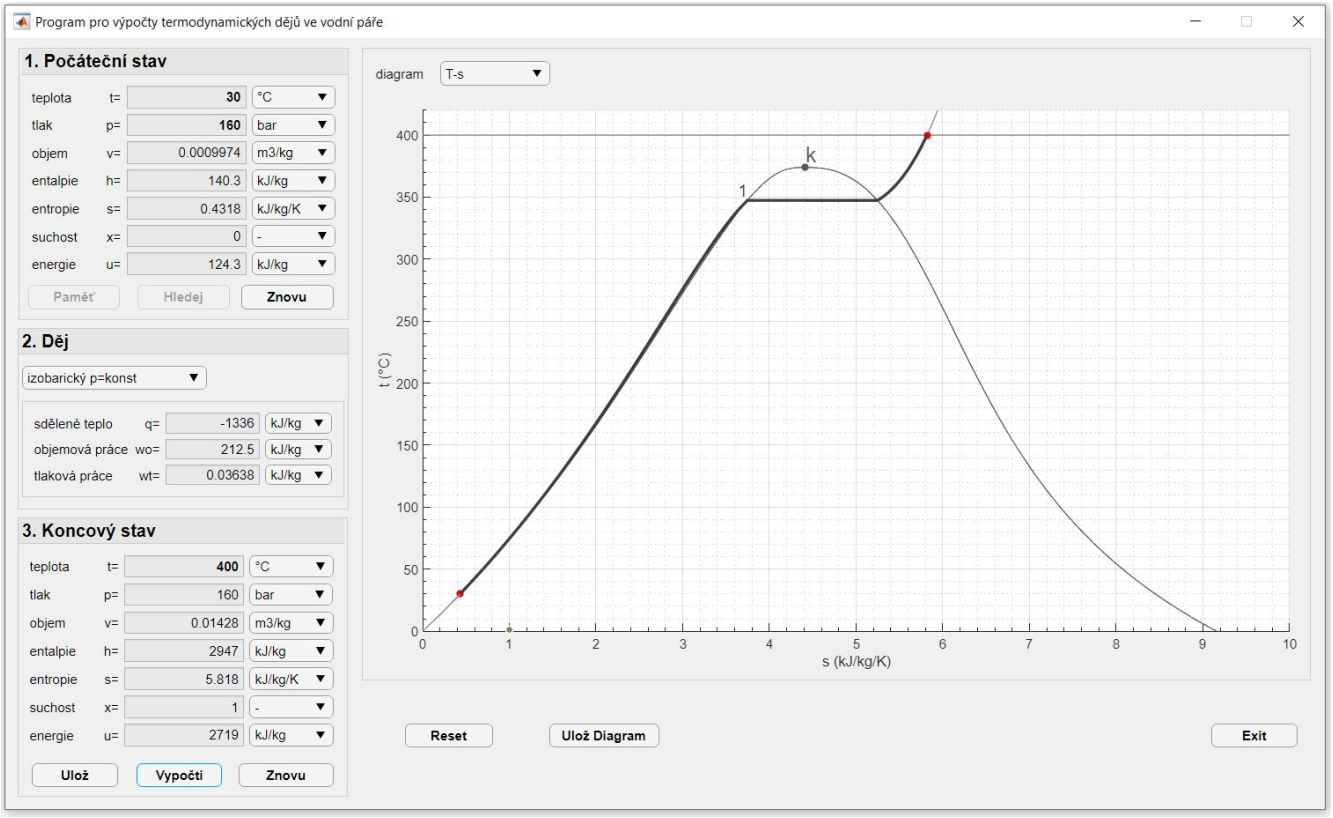
Při tvorbě programu byl kladen důraz na to, aby chování programu bylo intuitivní a nebylo možné program přivést do neočekávaného stavu. Určitá pole nejsou z počátku vidět. Na některá tlačítka z počátku nelze kliknout a některá editační pole nelze změnit. To může být zprvu překvapivé a proto je zde uveden příklad práce s programem.

Řekněme že je cílem spočítat izobarický ohřev vody z teploty $30\,°C$ na teplotu $400\,°C$ při tlaku $160\,bar$ a následná adiabatická expanze o účinnosti $0,8$ na teplotu $25\,°C$. Prvním krokem je spuštění programu. Následně je pak v okně programu zajímavý blok (kontejner) označený jako *Počáteční stav*. V tomto bloku jsou postupně vyplněna editační pole teploty a tlaku požadovanými hodnotami. Poté je možné toto zadání potvrdit stisknutím klávesy Enter (na klávesnici), kliknutím myší mimo poslední editované pole nebo přímo kliknutím na tlačítko *Hledej*. V libovolném z případů dojde k uzamčení editačních polí počátečního stavu. V každém případě je pak výpočet ostatních termodynamických veličin počátečního stavu proveden kliknutím na tlačítko *Hledej*.

Obrázek 7.7: Kontejner *Počáteční stav* před zadáním hodnot. Do editovacích polí je možné vpisovat.

Obrázek 7.8: Kontejner *Počáteční stav* po provedení výpočtu. Editovací pole jsou uzamčená.

Po stisku tlačítka *Hledej* je proveden výpočet a je odemčeno takzvané *dropdown menu* v bloku *Děj* a příslušná editační pole a tlačítka v bloku koncového stavu. Pak je ze seznamu dějů ve zmíněném dropdown menu možné vybrat požadovaný děj. Přednastavená hodnota tohoto menu je izobarický děj. Následuje vyplnění polí v bloku koncového stavu. Chování editačních polí koncového stavu je téměř stejné jako chování polí počátečního stavu. Rozdílem je, že je zde zadávána pouze jedna hodnota. V tomto demonstračním případě je zadána teplotu $400\,°C$. Potvrzením této hodnoty dojde k uzamčení všech editačních polí v bloku koncového stavu. Pak jsou stiskem tlačítka *Vypočti* vypočítané parametry děje a ostatní termodynamické veličiny koncového stavu. Také je vykreslen děj ve zvoleném diagramu. V základním nastavení programu v t-s diagramu. Okno programu pak v tomto stavu vypadá následovně:

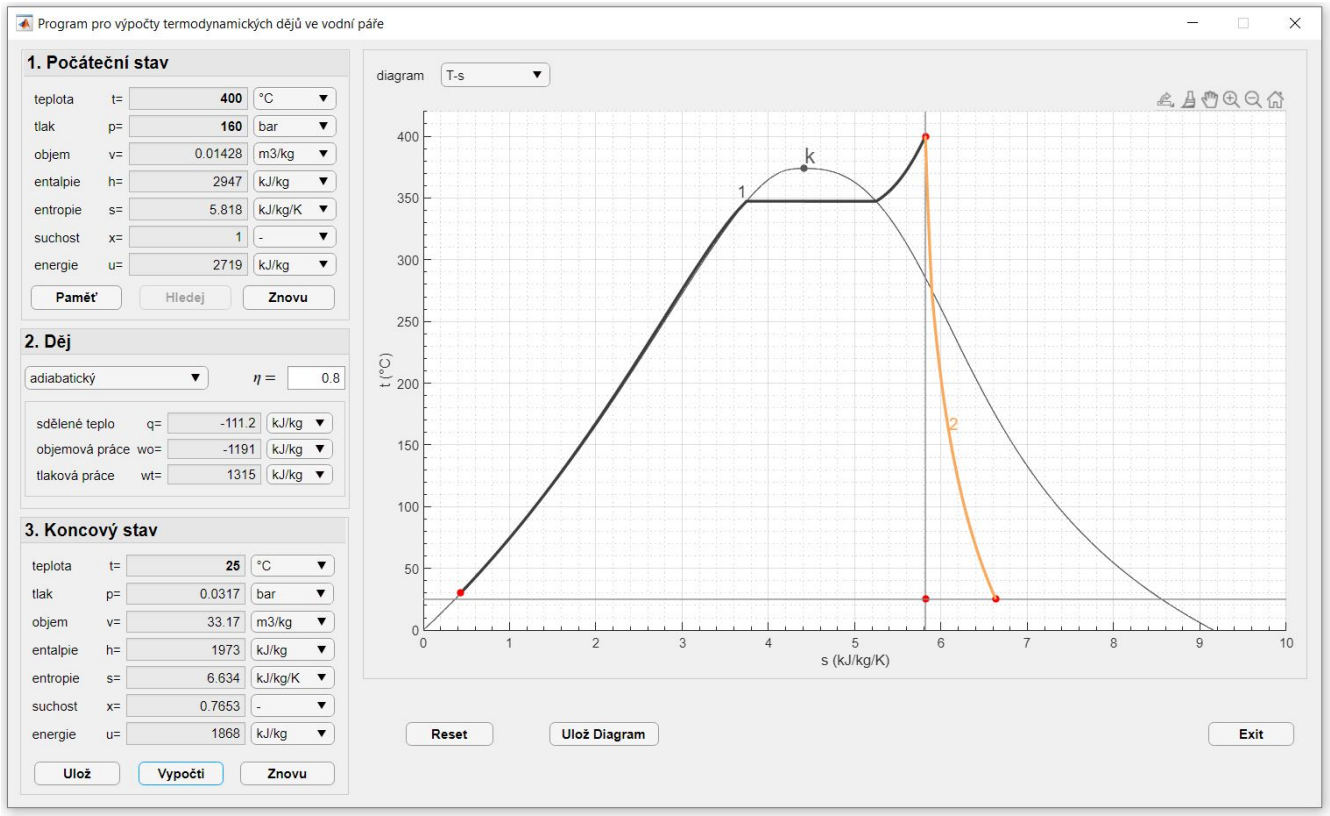


Obrázek 7.9: Okno programu ve stavu po výpočtu jednoho děje. Zobrazen t-s diagram.

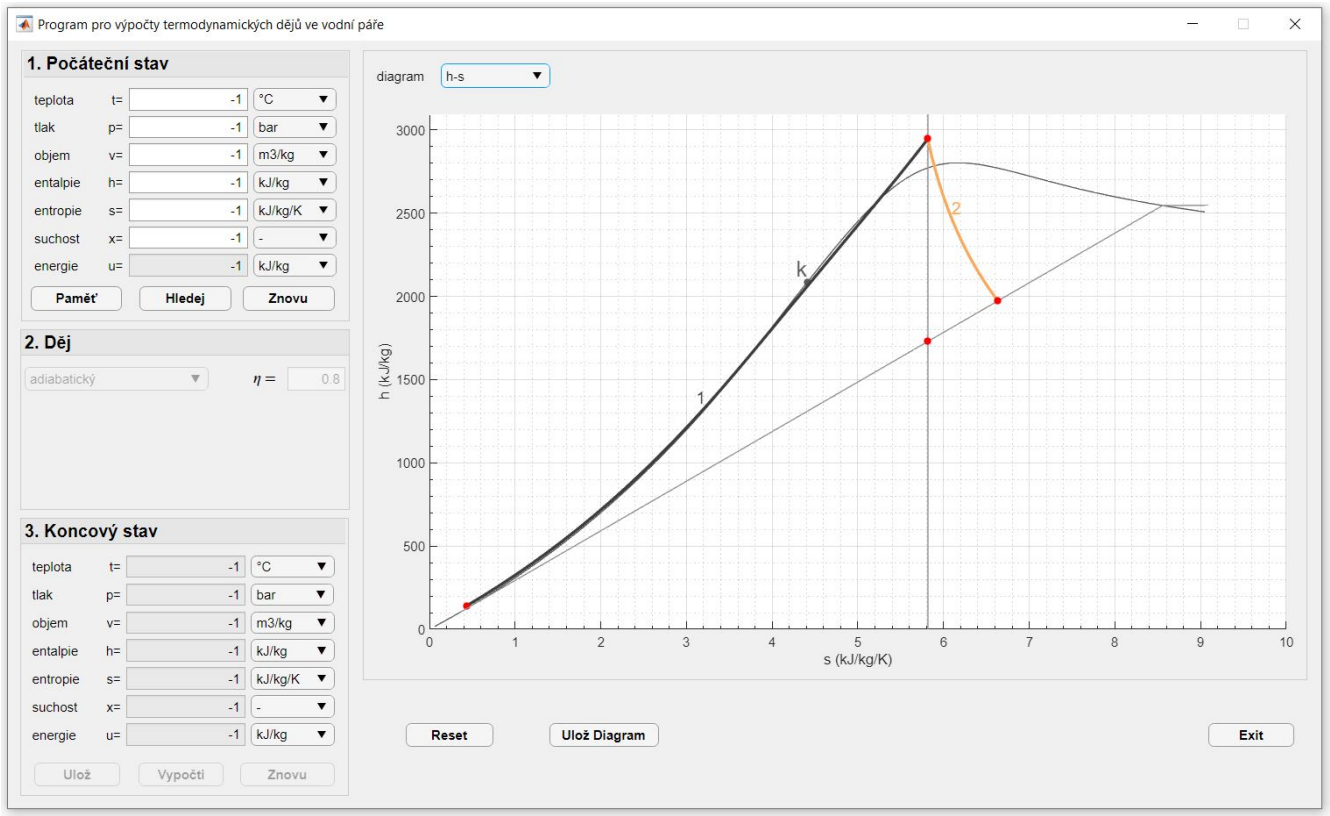
Pro uložení děje do paměti je pak stisknuto tlačítko *Ulož*. V tomto stavu v okně programu vidíme veličiny počátečního stavu, koncového stavu a parametry děje. Editační pole jsou uzamčená a není možné měnit děj. Odemčená jsou tlačítka *Paměť*, tlačítka rezetující stavy obslužná tlačítka a dropdown menu programu. Pro zadání dalšího děje je možné buď kliknout na příslušné tlačítko *Znovu* nebo pomocí tlačítka *Paměť* otevřít okno paměti. Pomocí tohoto okna je možné zobrazit libovolný uložený děj a pro účely dalších výpočtů načíst libovolný uložený stav jako počáteční stav. V tomto okamžiku jsou v paměti dva stavy. V základním nastavení programu je po otevření paměti předvybraný poslední uložený koncový stav. Výběr načtení stavu je potvrzen stiskem tlačítka *Ok*. Ukázka okna paměti 7.6 je zobrazena v sekci paměť programu.

Tím je do počátečního stavu zapsán vybraný stav. Hlavní okno programu je ve stavu jako po stisku tlačítka *Hledej* a tedy všechna editační pole v bloku počátečního stavu jsou uzamčená. Následuje výběr druhého děje. V případě této demonstrace adiabatický děj. Tedy je v dropdown menu vybrána položka adiabatického děje. To vede ke zobrazení editovatelného pole η. Pole je umístěné vedle dropdown menu výběru děje. Do tohoto pole je zadána účinnost daného adiabatického děje. Zadané číslo musí být mezi nulou a jedničkou. Tedy v případě procent je nutné hodnotu vydělit stem. Dále je zadána hodnota veličiny koncového stavu. Toto zadání je pak potvrzeno stiskem tlačítka *Vypočti*. Tím jsou opět dopočítány parametry děje, parametry koncového stavu a děj je zobrazen v t-s diagramu.

Pokud je žádoucí místo t-s diagramu zobrazit zadané (uložené) děje v jiném diagramu, tak je možné zobrazený diagram změnit pomocí dropdown menu umístěného nad diagramem. V případě demonstrace je přepnuto z t-s diagramu na h-s diagram. V tomto okamžiku nedošlo k uložení děje do paměti a tedy po změně diagramu vyskočí potvrzovací okno uložení. Potvrzením tohoto okna program uloží děj do paměti. V opačném případě je poslední vypočítaný děj ztracen. V obou případech dojde k překreslení diagramu a to včetně dějů uložených v paměti.

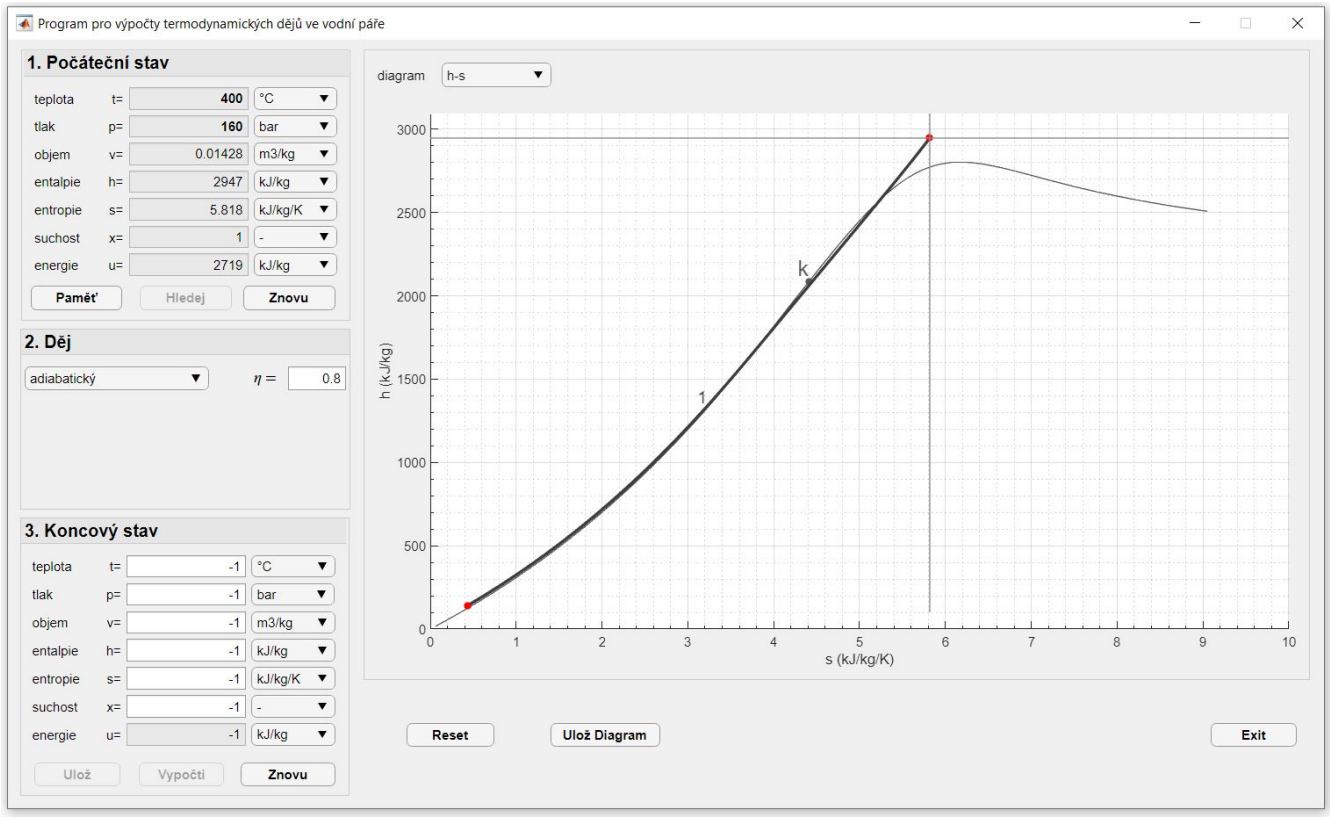


Obrázek 7.10: Okno programu ve stavu po výpočtu dvou dějů. Zobrazen t-s diagram.



Obrázek 7.11: Okno programu ve stavu po výpočtu dvou dějů. Zobrazen h-s diagram.

V některých případech může být žádoucí některé z uložených dějů z paměti programu odstranit. K tomu je možné využít okno paměti. Po jeho otevření stačí vybrat jeden z bodů děje, který je žádoucí vymazat a následně kliknout na tlačítko *Vymaž děj*. Tím je z paměti vybraný děj odstraněn a plocha diagramu je překreslena. Okno programu po odstranění děje pak vypadá takto:



Obrázek 7.12: Okno programu ve stavu po smazání posledního (adiabatického) děje. Zobrazen h-s diagram.

8. Závěr

Zadání této diplomové práce mělo celkem 3 body. První z úkolů byl provést rešerši na téma problematiky stavové plochy vodní páry a formulace standardu IAPWS-97. Rešerše stavové plochy vodní páry byla provedena především pomocí doporučené literatury. Rešerše také obsahuje informace o diagramech, jejich využití a to včetně dějů v nich znázorňovaných. Druhá část rešerše pak vychází z dokumentu mezinárodní organizace IAPWS. Tento dokument je standardem pro výpočty parametrů vodní páry v různých stavech. Druhá část rešerše obsahuje informace o dělení plochy vodní páry na oblasti. Toto dělení je dáno řešeními rovnic, které vychází z Gibbsovy bezrozměrové rovnice. Dále je uveden příklad definice jedné oblasti dle tohoto dokumentu.

Druhým bodem zadání bylo v prostředí Matlabu s využitím knihovny XSteam vytvořit skripty pro tisk t-s a h-s diagramů. V rámci tohoto úkolu byly vytvořeny funkce pro výpočet bodů na izočárách v plochách h-s a t-s diagramu. Dále byl vytvořen script pro tisk těchto dvou diagramů s nastavením pro tisk na formát papíru A3. Škálované zmenšeniny (formát A4) těchto diagramů jsou k přiložené k této práci.

Třetím bodem zadání bylo s využitím knihovny XSteam vytvořit v prostředí Matlabu program pro řešení termodynamických dějů ve vodní páře. Pro vytvoření grafického rozhraní hlavního těla programu bylo využito prostředí Matlabu App Designer. V rámci tohoto prostředí byla vytvořena většina logiky fungování uživatelského rozhraní. Výpočetní funkce a okno paměti programu pak byly vytvořeny přímo jako funkce prostředí Matlabu. Program je vybaven funkcemi pro vykreslení h-s, t-s a p-v diagramů do kterých následně vykresluje uživatelem zadané izočáry (pomocné čáry) a termodynamické děje. Program je vybaven pamětí dějů ze které je možné načítat jednotlivé stavy a zvolené děje cíleně vymazávat. Program dále pro další snadné zpracování parametrů děje a koncových stavů umožňuje tyto veličiny kopírovat pomocí schránky do jiných programů, například Excel. Dále program umožňuje zobrazený uložit diagram na disk.

Program umožňuje výpočty izobarického děje (p = konst.), izotermického děje (t=konst.), izochorického děje (v=konst.), izoentropického děje (s=konst.), izoentalpického děje (h=konst.) a adiabatického děje (q=0) s uživatelem zadanou účinností. Pro definici počátečního stavu program uživateli umožňuje v rozmezí hodnot daných možnostmi knihovny XSteam zadat libovolnou dvojici stavových veličin. Pro vybraný termodynamický děj z výše uvedené nabídky a jednu stavovou veličinu program umožňuje dopočítat ostatní hodnoty koncového stavu. V rámci vypočtených termodynamických veličin pak program k veličinám, které je možné zadat, přidává vnitřní energii (u).

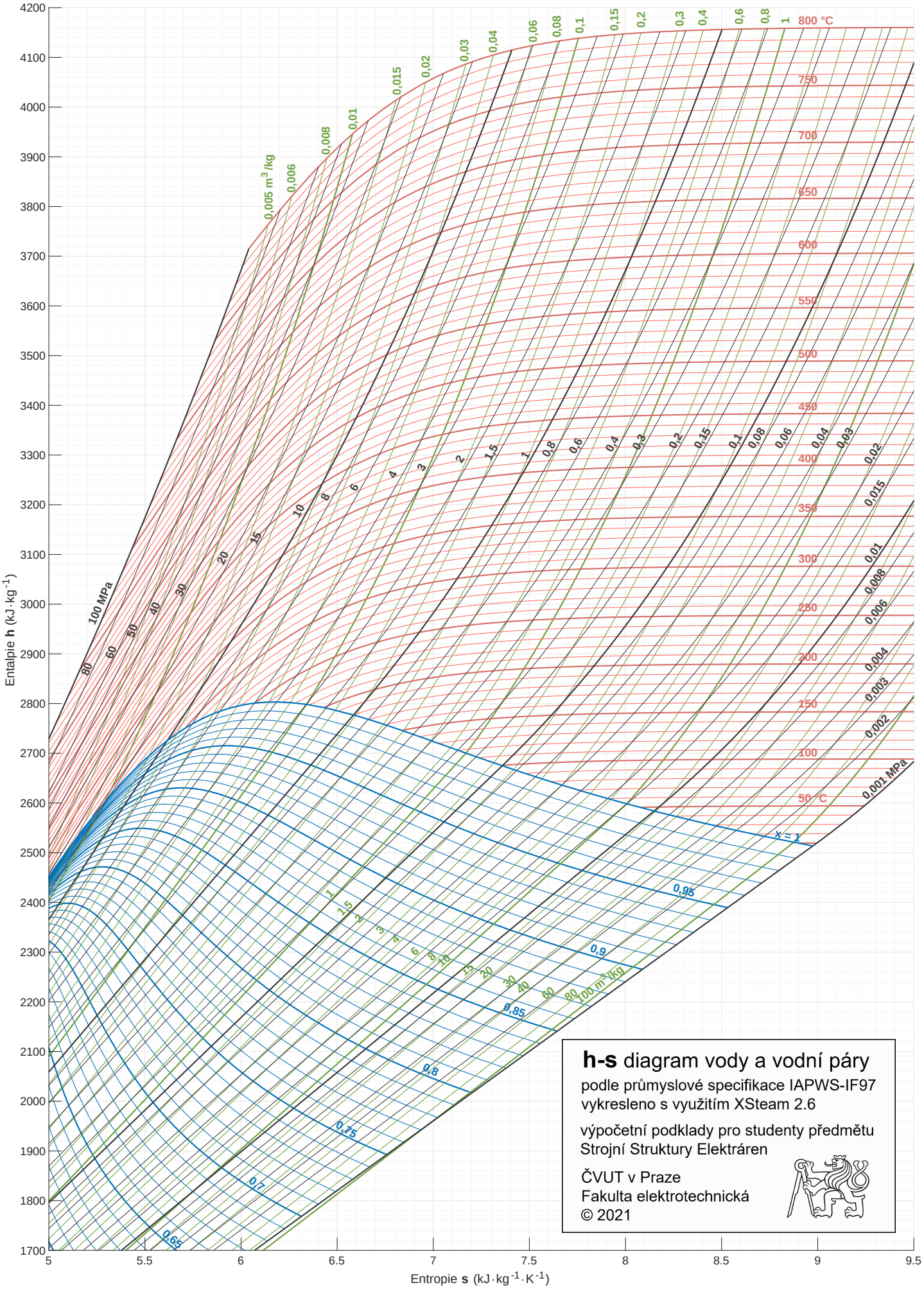
Zadání tedy bylo splněno v plném rozsahu.

9. Použité zkratky

IFC-67	The 1967 IFC Formulation for Industrial Use
IAPWS	The International Association for the Properties of Water and Steam
IAPWS IF97	IAPWS Industrial Formulation 1997
IAPWS-95	IAPWS Formulation 1995 for the Thermodynamic Properties of Ordinary Water Substance for General and Scientific Use
h	entalpie
s	entropie
t	teplota
p	tlak
v	objem
x	suchost
u	vnitřní energie
ps	počáteční stav
ks	koncový stav
q	sdělené teplo
w_o	objemová práce
w_t	tlaková práce

10. Přílohy

1. h-s diagram
2. t-s diagram

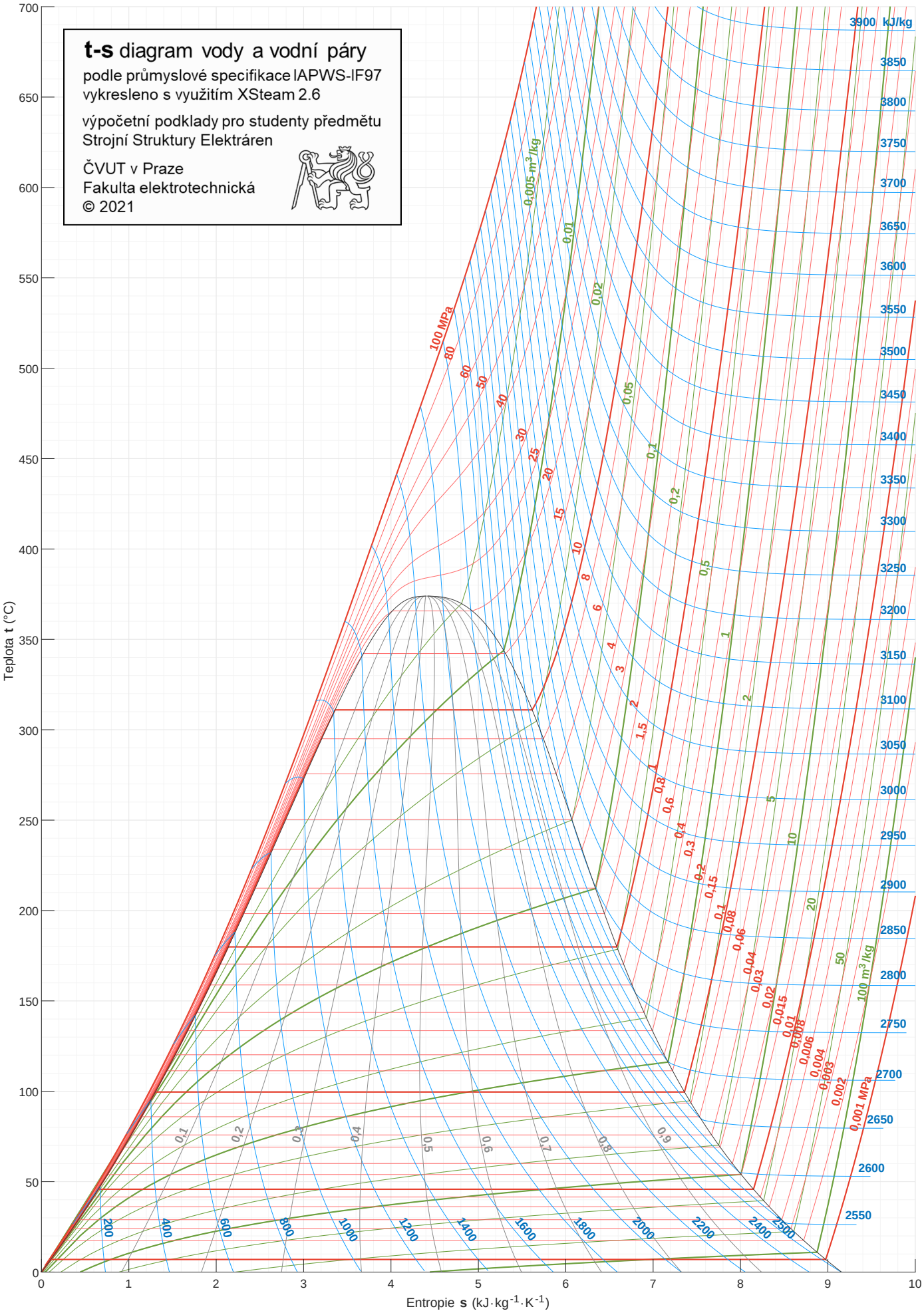


h-s diagram vody a vodní páry

podle průmyslové specifikace IAPWS-IF97
vykresleno s využitím XSteam 2.6

výpočetní podklady pro studenty předmětu
Strojní Struktury Elektráren

ČVUT v Praze
Fakulta elektrotechnická


t-s diagram vody a vodní páry
podle průmyslové specifikace IAPWS-IF97
vykresleno s využitím XSteam 2.6
výpočetní podklady pro studenty předmětu
Strojní Struktury Elektráren
ČVUT v Praze
Fakulta elektrotechnická
© 2021
Teplota t (°C)
Entropie s (kJ·kg-1·K-1)
3900 kJ/kg
100 MPa
0,005 m3/kg
0,001 MPa
100 m3/kg

Bibliografie

1. *FAQ: ASME Steam Tables* [online]. [B.r.]. [cit. 2022-04-20]. Dostupné z: http://www.iapws.org/faq2/asmetab.html.
2. M.PAVELEK, E.JANOTKOVÁ. *Návod k obsluze software PÁRA*. Brno, Technická 2, 2002.
3. CENGEL, Yunus A.; BOLES, Michael A. *Thermodynamics: an engineering approach*. 5th. New York: McGraw-Hill, 2006. ISBN 0073529214;9780071250849;9780073529219;0071250840;
4. ŠAFAŘÍK, Pavel; VESTFÁLOVÁ, Magda. *Termodynamika vlhkého vzduchu*. 1. vydání. Praha: České vysoké učení technické v Praze, 2016. ISBN 9788001060209;8001060209;
5. *Kurz: Strojní struktury elektráren - B201* [online]. [B.r.]. [cit. 2022-04-20]. Dostupné z: https://moodle.fel.cvut.cz/course/view.php?id=5418.
6. ZAPLATÍLEK, Karel; DOŇAR, Bohuslav. *MATLAB: tvorba uživatelských aplikací*. 1. vyd. Praha: BEN - technická literatura, 2004. ISBN 9788073001339;8073001330;
7. GILAT, Amos. *MATLAB: an introduction with applications*. 3rd. Hoboken: Wiley, 2008. ISBN 9780470108772;0470108770;
8. *Develop Apps Using App Designer - MATLAB & Simulink* [online]. [B.r.]. [cit. 2022-04-23]. Dostupné z: https://www.mathworks.com/help/matlab/app-designer.html.
9. *International Association for the Properties of Water and Steam* [online]. [B.r.]. [cit. 2022-04-20]. Dostupné z: http://www.iapws.org/.
10. WAGNER, W.; COOPER, J. R.; DITTMANN, A.; KIJIMA, J.; KRETZSCHMAR, H.-J.; KRUSE, A.; MARES˘, R.; OGUCHI, K.; SATO, H.; STO¨CKER, I.; S˘IFNER, O.; TAKAISHI, Y.; TANISHITA, I.; TRU¨BENBACH, J.; WILLKOMMEN, Th. The IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam. *Journal of Engineering for Gas Turbines and Power* [online]. 2000, roč. 122, č. 1, s. 150–184 [cit. 2022-04-20]. ISSN 0742-4795, ISSN 1528-8919. Dostupné z DOI: 10.1115/1.483186.
11. MAGNUS HOLMGREN. *XSteam license*. 1999.
12. *X Steam, Thermodynamic properties of water and steam.* [online]. [B.r.]. [cit. 2022-04-03]. Dostupné z: https://www.mathworks.com/matlabcentral/fileexchange/9817-x-steam-thermodynamic-properties-of-water-and-steam.
13. *XSteamW - a vectorizing wrapper for XSteam* [online]. [B.r.]. [cit. 2022-04-16]. Dostupné z: https://www.mathworks.com/matlabcentral/fileexchange/29186-xsteamw-a-vectorizing-wrapper-for-xsteam.
14. *Graphics renderer information - MATLAB rendererinfo* [online]. [B.r.]. [cit. 2022-04-17]. Dostupné z: https://www.mathworks.com/help/matlab/ref/rendererinfo.html.
15. *Control appearance and behavior of figure window - MATLAB* [online]. [B.r.]. [cit. 2022-04-17]. Dostupné z: https://www.mathworks.com/help/matlab/ref/matlab.ui.figure-properties.html#property_d119e283913.