Univerzita Palackého v Olomouci

Katedra experimentální fyziky

Bakalářská práce

Zpracování experimentálních dat z oblasti atomové fyziky v programu *Mathematica*

Autor	Pavla Šretrová
Vedoucí práce	Mgr. Jan Říha, Ph.D.
Studijní obor	Aplikovaná fyzika
Forma studia	Prezenční
Rok obhajoby	2014

Bibliografická identifikace

Autor	Pavla Šretrová
Název práce	Zpracování experimentálních dat z oblasti atomové fyziky v programu *Mathematica*
Typ práce	Bakalářská
Pracoviště	Katedra experimentální fyziky
Vedoucí práce	Mgr. Jan Říha, Ph.D.
Rok obhajoby	2014
Počet stran	52
Počet příloh	3
Jazyk	Český
Abstrakt	Soubor dat získaný z desítek měření experimentu jaderného dopředného rozptylu (NFS, *Nuclear Forward Scattering*) byl zpracován v symbolickém výpočetním programu *Mathematica*. Nestandardní podoba dat vyžadovala jejich postupnou úpravu a nalezení vhodného algoritmu pro zpracování dat jednoho měření. Tento postup se zobecnil a aplikoval na všechna měření. Výsledkem této práce jsou interaktivní výstupy grafů a tabulek, které se budou dále zpracovávat ve výzkumu oxidačních vlastností vysokovalenčních stavů železa.
Klíčová slova	*Mathematica*, zpracování dat, programování

Bibliographic identification

Author	Pavla Šretrová
Title	Processing experimental data in the field of athomic physics using software *Mathematica*
Type of thesis	Bachelor
Department	Department of Experimental Physics
Supervisor	Mgr. Jan Říha, Ph.D.
Year of presentation	2014
Number of pages	52
Number of appendices	3
Language	Czech
Abstract	Data set obtained from dozens of experimental measurements of nuclear forward scattering was processed in a symbolic computing program called *Mathematica*. Uncommon data form required gradual treatment and finding of suitable algorithm for data processing of one measurement. This process was generalized and applied on all measurements. Results of this work are interactive outputs in form of graphs and tables. Those outputs will be further processed in research of higher oxidation state of iron.
Key words	*Mathematica*, data processing, programming

Prohlašuji, že jsem bakalářskou práci vypracovala samostatně a že jsem použila zdrojů, které cituji v seznamu literatury.

V Olomouci dne …… ……..

podpis

Mnohokrát děkuji vedoucímu bakalářské práce Mgr. Janu Říhovi, Ph.D. za veškerou poskytnutou pomoc, za cenné rady a za čas, který mi věnoval při tvorbě této práce. Dále bych chtěla poděkovat Mgr. Vítu Procházkovi, Ph.D. a Mgr. Davidu Smrčkovi za jejich odborné rady a věcné připomínky. V neposlední řadě také děkuji své rodině za jejich podporu během mého studia.

Obsah

1. Úvod

Zpracování dat, úprava složitých výrazů a vizualizace získaných výsledků jsou neoddělitelnou součástí každodenní práce vědců, výzkumných pracovníků, studentů vysokých i středních škol, technických odborníků a jiných. Společnost Wolfram Research, Inc. poskytuje k usnadnění a zrychlení jejich práce mocný nástroj v podobě symbolického výpočetního softwaru *Mathematica*.

Tento program upřednostňuje symbolické výpočty před numerickými, výpočet tedy probíhá postupnou úpravou *výrazů*. Dalo by se říci, že *Mathematica* postupuje při počítání stejně jako člověk. Mezi další silné stránky tohoto programu patří interaktivní výstupy, dynamické výstupy v podobě modelování a simulací, kompatibilita s jinými programy pomocí protokolu *MathLink* a také kompletní programovací prostředí, které zahrnuje paradigmata všech hlavních programovacích stylů (procedurální, funkcionální, rule - based, objektové a další).

Cílem této práce bylo zpracovat rozsáhlý soubor dat v programu *Mathematica*. Data byla naměřena v Hamburském výzkumném centru DESY na synchrotronu PETRA III. Počet vzorků, na kterých probíhal jaderný dopředný rozptyl, se pohybovalo okolo 13, avšak vzorky se proměřovaly opakovaně za různých podminek. Celkový počet experimentů se tedy vyšplhal na 55. Během každého experimentu se na jednom počítači zaznamenávala teplota v reálném čase a na druhém se zapisovala v minutových intervalech intenzita záření, které prošlo vzorkem. Naším úkolem tedy bylo přiřadit každému takovému intervalu odpovídající časový interval.

Stručná charakteristika experimentu jaderného dopředného rozptylu je společně s podrobným popisem skladby dat uvedena v kapitole 2. Kapitola 3 je zaměřená na stručnou historii a strukturu softwaru. Dále pak zahrnuje popis tří základních programovacích stylů, které *Mathematica* podporuje. Kapitola 4 obsahuje popis algoritmu zpracování dat. Nejdříve je důkladně okomentován postup pro jedno měření a v další části této kapitoly je uveden komentář postupu zpracování všech měření.

Hlavní literatura, ze které jsme čerpali, je od autorů P. Wellina, R. Gaylorda a S. Karmina - *An Introduction to Programming with Mathematica* (třetí vydání). Tato kniha je podrobným návodem, který je určený širokému spektru uživatelů od úplných začátečníku, až po pokročilé uživatele. Dále jsme používali také *Documentation Center*, což je průvodce, který je součástí programu *Mathematica*. Obsahuje veškeré zabudované funkce programu a poskytuje rozsáhlé návody a názorné příklady.

2. Popis experimentu NFS

2.1 Teorie

2.1.1 Jaderný rezonanční rozptyl

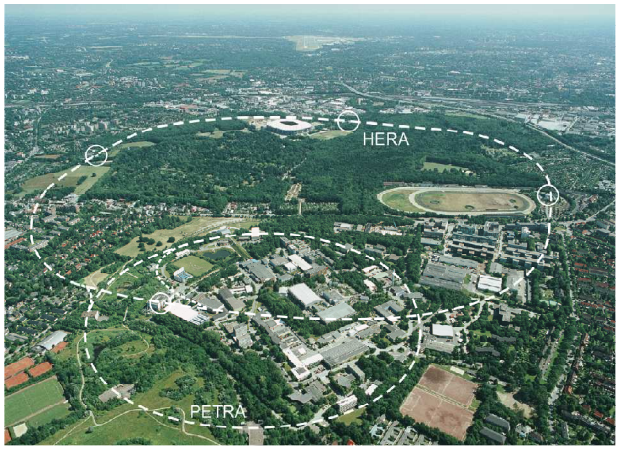
Jaderný rezonanční rozptyl (NRS, Nuclear Resonance Scattering) je možné dělit podle dvou kritérií, která jsou elasticita a koherence. Běžně využívanými procesy jsou koherentní elastický (NFS a NBS) a neelastický jaderný rozptyl (koherentní i nekoherentní). Metoda NRS, někdy také označovaná jako synchrotronová Mössbauerova spektroskopie, našla významné uplatnění ve fyzice pevných látek. Umožňuje například studium hyperjemných interakcí, hyperjemných polí nebo také měření fononových spekter. Jedná se o rozptyl elektromagnetického záření na jádrech, která jsou charakterizována hmotností m, číslem protonovým Z (tedy i nábojem jádra $q = Z \cdot e$) a nukleonovým A, kvadrupólovým momentem Q, spinem s (různý pro základní a excitovaný stav), magnetickým moment m a paritou $\prod$.

Atomové jádro se může nacházet pouze ve stavu s danou energií. Tyto stavy se označují jako stacionární. Každý má jinou hodnotu energie, kde stav s nejnižší energií je základní. Přechod mezi jednotlivými stacionárními stavy je možný pouze pro přesně definované podmínky za současného vyzáření nebo pohlcení kvanta elektromagnetického záření. Během tohoto procesu platí zákony zachování energie, hybnosti, momentu hybnosti a parity [1].

Uvažujme emisi fotonu jádrem, které má před vyzářením fotonu nulovou hybnost. Tento foton má však určitou rychlost a tedy i hybnost. Dle zákona zachování hybnosti musí mít jádro stejně velkou hybnost opačného směru, tento jev se označuje jako zpětný ráz. Pokud je však atomové jádro vázáno v krystalové mřížce, přechází hybnost na krystal jako celek. Výsledkem je mimořádně úzká čára jaderného přechodu, řádově 10^{-10} až 10^{-5} eV. Výše popsaný jev poprvé objasnil roku 1958 německý fyzik Rudolf Mössbauer, podle něhož se bezodrazová emise nazývá. Analogicky je známá také bezodrazová absorpce [2, 3].

Pro experimenty NRS se využívá synchrotronové záření. Je to pulzní elektromagnetické záření generované synchrotrony, které vzniká při kruhovém pohybu částic jako důsledek konečné rychlosti světla [4]. Toto záření bylo poprvé pozorováno v roce 1947. Pro každý experiment je zapotřebí záření se specifickými vlastnostmi, jako jsou napří-

klad briliance, světlost (brightness) nebo polarizace [1]. Mezi významné synchrotrony, ve kterých jsou možné experimenty NRS, patří Hamburské centru DESY (*German Electron Synchrotron*) [5], ANKA [6], ESRF (*European Synchrotron Radiation Facility*) situovaném ve francouzském Grenoblu [7], SPring8 v Japonsku [8] nebo také APS (*Advance Photon Source*) [9].



Obr. 1: Podzemní synchrotrony HERA a PETRA v Hamburském centru DESY. Zdroj http://www-zeus.desy.de/~bellagam/figure.html.

2.1.2 Koherentní elastický rozptyl

NRS experimenty se vyznačují tím, že jádra po rozptylu jsou ve stejném stavu jako před rozptylem, nedochází tedy ke změně jaderného systému. Také celková hodnota energie samotného záření zůstává nezměněná.

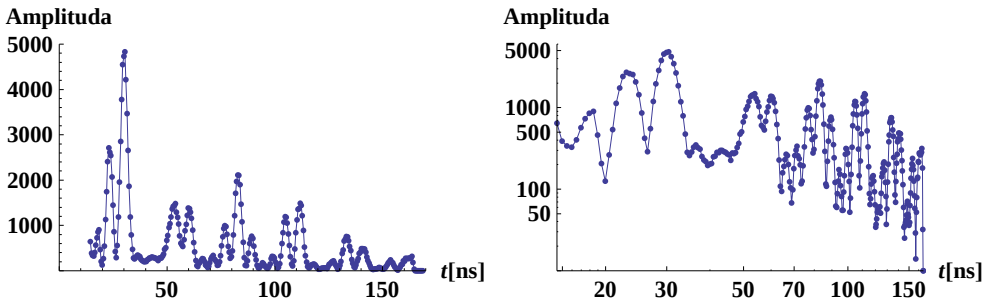
Jsou známé dva typy koherentního elastického rozptylu. Rozptyl dopředný (NFS, Nuclear Forward Scattering), kdy je rozptýlené záření detekované v přímém směru, a rozptyl Braggův (Nuclear Bragg Scattering). Při tomto rozptylu dochází k difrakci dopadajícího záření, které je pak detekováno pod různými úhly odpovídajícími Braggovu zákonu

$$n\lambda = 2d_{hkl}\sin\Theta.$$

Každý vzorek obsahuje určité množství izotopu jednoho prvku. Například přirozená směs železa obsahuje 2 % izotopu ^{57}Fe, který má energetický rozdíl základního a excitovaného stavu přibližně 14,4 keV. Dopadá-li elektromagnetické záření o této energii na systém, část projde vzorkem bez interakce, část interaguje s elektrony v elektronovém obalu nebo fonony a část je absorbována atomovými jádry. Ty přejdou do prvního excitovaného stavu, který má určitou dobu života. Následně pak jádra

relaxují zpět do základního stavu. Současně vyzáří kvantum elektromagnetického záření, které je zpožděné za nerozptýleným zářením právě o dobu setrvání v excitovaném stavu. Zpoždění záření interagujícího s elektrony a fonony je zanedbatelné.

Spektrum jaderného dopředného rozptylu chápeme jako závislost tohoto záření na zpoždění za excitačním pulzem. Na obrázku 2 je uveden příklady spektra NFS v lineární a logaritmické škále [1].



Obr. 2: Ukázka spektra jaderného dopředného rozptylu v lineární (vlevo) a logaritmické (vpravo) škále pro práškový vzorek Fe_3O_4.

Jen některé izotopy splňují podmínky pro experimenty NFS. První z nich je vhodná energie přechodu mezi základním a excitovaných stavem, která musí mít takovou hodnotu, jakou lze získat z elektromagnetického záření synchrotronu. Dále musí mít nenulový magnetický moment *m*, který interaguje s elektrickým i magnetickým polem. Izotop musí být stabilní nebo by měl mít alespoň takový poločas rozpadu, který je dostatečně dlouhý v porovnání s délkou života excitovaného stavu jádra. Nejčastěji používanými prvky, kdy alespoň jeden izotop splňuje výše uvedená kritéria, patří Fe, Eu, Ge, Sn, Sm, Ta. Nejvýznamnějším z nich je již zmiňovaný izotop ^{57}Fe.

V porovnání s klasickou Mössbauerovou spektroskopií je čas načtení spektra z NFS mnohem kratší, řádově v desítkách sekund. Tím se otevírají dveře ke studiu rychle probíhajících fyzikálních procesů jako je difúze, krystalizační a rekrystalizační procesy, chemické reakce a další.

2.2 Realizace experimentu NFS

Měření NFS proběhlo v říjnu roku 2013 v Hamburském centru DESY v Německu. Bylo použito synchrotronového záření kruhového urychlovače PETRA III. Parametry záření jsou uvedeny v tabulce 1.

Vlastnost/veličina	*Hodnoty synchrotronového žáření*
Brightness (ph/s/eV/sr)	$2.8 \cdot 10^{22}$
Brilliance (ph/s/eV/sr/mm^2)	$2.8 \cdot 10^{22}$
Rozměr svazku (mm^2)	1×2.5
Energie svazku (keV)	14.41
Šířka energetického spektra (meV)	5
Polarizace	lineární v rovině prstence

Tab.1: Tabulka s parametry synchrotronového záření urychlovače PETRA použitého pro měření experimentů NFS v říjnu 2013.

V centru DESY jsou experimenty řízené počítačem s OS Linux. Z časových, technických a organizačních důvodů nebylo možné ovládací program pícky, která byla odzkoušená v České Republice na separátním počítači s OS Windows, přizpůsobit a nainstalovat do řídícího počítače. Tento problém byl vyřešen použitím dvou počítačů. Jednoho pro řízení pícky a druhého pro řízení celého experimentu. Jako nejjistější kontrola synchronizace programů se zvolil reálný čas s dostatečnou přesností +/- 1s. Na jednom počítači se zapisovala teplota do textových souborů (*.txt). Na druhém se měřila spektra NFS s časovou závislostí, která se zaznamenávala do souborů s příponou *.fio. Měření spekter NFS probíhalo v minutových intervalech. Mezi těmito intervaly jsou několikasekundové pauzy, které jsou způsobeny spouštěním akumulace dat.

2.2.1 Postup zpracování dat

Máme tedy jeden soubor (*.txt) s teplotami a k němu několik souborů (*.fio) s informacemi o časově závislých spektrech. Cílem je rozdělit soubor s teplotami na časové intervaly. Koncové časy intervalů jsou časy zápisu spektrálních dat v *.fio souborech. Odečtením 60 s získáme počáteční časy intervalů. Tyto časy porovnáme s časy v souboru teplot a separujeme jednotlivé intervaly. Každý takový interval zpracujeme a zapíšeme do tabulky, která bude obsahovat souhrnně maximální, minimální a průměrnou teplotu, název přislušného fio souboru, počáteční a konečný čas měření. Na závěr vykreslíme graf závislosti teploty na čase pro jednotlivá měření. Výše popsaný postup zrealizujeme v programu *Mathematica*.

3. Mathematica

3. 1 Historie

První verze programu *Mathematica* byla zveřejněná 23. června roku 1988. Jejím autorem je Stephen Wolfram, teoretický fyzik, který hledal v oblasti počítačové techniky nástroj pro usnadnění a zrychlení matematických výpočtů.

Historie programu *Mathematica* sahá až do roku 1981, kdy Wolfram představil svůj první symbolický výpočetní systém *SPM* (*Symbolic Manipulation Program*). Byl předchůdcem programu *Mathematica*. Oba tyto programy měli společnou myšlenku: "*vše je symbolický výraz*".

První kód programu *Mathematica* byl napsán v říjnu 1986 a v polovině roku 1987 byl připravený. V dubnu 1988 byla hotová první verze *Mathematicy* 1.0 a 23. června byla představena veřejnosti pod záštitou firmy Wolfram Research, Inc., jejímž generálním ředitelem se stal právě Wolfram [10].

Mathematica byla neustále vyvíjena a doplňována o další užitečné nástroje jako je *MathLink, Wolfram Mathematica Player* (přehrávač souborů napsaných v programu *Mathematica*) nebo databáze *Wolfram Research*. Zde jsou obsažena nejaktuálnější data z oblasti matematiky, geografie, techniky, ekonomiky, financí a lingvistiky. Tato databáze vznikla jako online rozšíření verze *Mathematica* 6.0 a stala základním kamenem univerzálního vyhledávajícího výpočetního nástroje *Wolfram|Alpha*, který obsahuje veškeré dostupné algoritmy a znalosti [11].

V průběhu 26 let bylo vytvořeno přes 15 verzí programu *Mathematica*. Přehled hlavních verzí včetně jejich data zveřejnění je vypsán v tabulce 2. Nejnovější verze, *Mathematica* 9.0, byla zveřejněna 28.11.2012. Významným doplněním této verze je interaktivní input asistent, který automaticky dopňuje kód během jeho sázení, poskytuje šablony kódů, usnadňuje orientaci pomocí dynamického zvýrazňování. Také umožňuje zobrazit přímo v *notebooku* informace o funkci a jejích vlastnostech. Novým užitečným doplňkem je také *Suggestions Bar*, tzv. lišta s návrhy dalších úprav výstupu, pod kterým se ihned po vyhodnocení objevuje. Další novinkou je jednotkový systém, analýza sociální sítě, kompletní podpora náhodných procesů a další [12].

Verze	Datum zveřejnění	Verze	Datum zveřejnění
1.0	23.06.1988	4.2	01.11.2002
1.1	31.10.1988	5.0	12.06.2003
1.2	01.08.1989	5.1	25.11.2004
2.0	15.01.1991	5.2	20.06.2005
2.1	15.06.1992	6.0	01.05.2007
2.2	01.06.1993	7.0	18.11.2008
3.0	03.09.1996	8.0	15.11.2010
4.0	19.05.1996	9.0	28.11.2012
4.1	02.11.2000		

Tab. 2: Přehled hlavních verzí Mathematiky s daty jejich zveřejnění. Převzato z [13] a upraveno.

3. 2 Struktura

Mathematica je modulární softwarový systém, který má 2 hlavní části:

- *Kernel*, neboli *jádro*, je srdcem celé *Mathematicy*. Provádí veškeré výpočty, které jsou zadané jako vstupy (*input*) ať už z *frontendu* nebo z jakéhokoliv jiného programu pomocí *MathLink*.
- *Frontend* je uživatelské prostředí, které zprostředkovává komunikaci mezi uživatelem a *jádrem*. Vizualizace *frontendu* je řešená interaktivním dokumentem *notebook* (*.nb).

Jádro a *frontend* jsou od sebe oddělené. Taková konstrukce systému poskytuje více možností na rozdíl od celistvých systémů. *Frontend* může být spuštěn na lokálním počítači s rozšířenou grafikou, zatímco jádro může běžet na vzdáleném rychlejším počítači, nebo z jednoho *frontendu* může být spuštěno více *jader*, což nám umožní provádět paralelní výpočty.

Nejběžnějším způsobem manipulace s programem *Mathematica* je *notebook*. Jeho struktura je dána posloupností *buněk* (*cells*), kde každá *buňka* je buď vstup nebo výstup grafiky, zvuku, matematických výrazů a samožřejmě jejich kombinací, souhrnně označovaných jako *výrazy* (*expressions*). *Notebook* lze použít pro numerické, symbolické i grafické výpočty nebo také jako prostředek pro prezentaci a publikaci našich výsledků.

Mathematica používá kromě interaktivního rozhraní (*notebook*) další dva typy rozhraní, textové (*text - based*) a *MathLinks*.

MathLink je protokol používaný k vzájemné komunikaci programu *Mathematica* s externími programy na vysoké úrovni. Umožňuje výměnu *výrazů* mezi programy, které běží na stejných nebo různých počítačích, dokonce i v případě, že je počítačová

síť různorodá. Tento protokol je součástí programu Mathematica. Dále je možné ho využít v počítači jako přenosnou knihovnu programovacího jazyka C [14, str. 190] [15, str. 24].

3. 3 Programování

Nejběžnějšími datovými typy, které se používají v programování, jsou čísla, textové řetězce, symboly a *Listy*. List je speciální datovou strukturou ve tvaru

$$\{e_1, e_2, \ldots, e_n\}$$

Seskupuje jednotlivé *výrazy* $\{e_1, e_2, \ldots, e_n\}$ do jednoho celku, jako jsou například:

- vektory

```
In[1]:= v = {0, y - 1, z^2};
In[2]:= VectorQ[v]
Out[2]= True
```

- matice

```
In[3]:= mat = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
In[4]:= MatrixQ[mat]
Out[4]= True
```

- množiny

```
In[5]:= A = {α, β, γ, δ, ϵ};
        B = {α, γ, ξ, κ, μ};
In[7]:= A ⋂ B
Out[7]= {α, γ}
In[8]:= A ⋃ B
Out[8]= {α, β, γ, δ, ϵ, ξ, κ, μ}
```

V programu *Mathematica* výraz, neboli *exppression*, představuje jakoukoliv strukturu. Znamená to, že datový objekt, grafika, jednoduchý výpočet, buňka v *notebooku*, i *notebook* samotný se označuje jako *výraz*. Pochopení této myšlenky je nezbytné pro efektivní práci se softwarem [15, str. 31].

Programování v programu *Mathematica* se neobejde bez přiřazení jednoho výrazu druhému. Symbolická struktura softwaru podporuje dva typy přiřazení (*assignment*). Okamžité (*immediate*) a opožděné (*delayed*).

```
LHS = RHS
LHS := RHS
```

Okamžité přiřazení výraz na pravé straně nejdříve dosadí a následně vyhodnotí. U zpožděného přiřazení se výraz pouze zadefinuje, ale nevyhodnotí. Vyhodnotí se až v případě použití definice. Pro lepší pochopení uvádíme názorný příklad rozdílu mezi těmito dvěma přiřazeními.

```
In[9]:= rand1 = RandomReal[];

In[10]:= rand2 := RandomReal[];

In[11]:= Table[rand1, {3}]
         Table[rand2, {3}]

Out[11]= {0.544977, 0.544977, 0.544977}

Out[12]= {0.773958, 0.173249, 0.143964}
```

Jak jde vidět, u opožděného přiřazení se výraz na pravé straně vyhodnotil pokaždé, když jsme použili jeho definici. Proto jsme odbrželi tři různá náhodná čísla. Avšak u okamžitého přiřazení se výraz na pravé straně nejdříve vyhodnotil a výsledek byl přiřazen symbolu **`rand1`**. Výsledkem je tedy *list* tří stejných čísel [16].

Mathematica podporuje celou škálu programovacích stylů jako jsou procedurální, funkcionální, grafické, rule-based, rekurze a jiné. Budeme zde diskutovat jen některé z nich.

3. 3. 1 Procedurální (příkazové) programování

Procedurální programování je styl programování, který je charakterističtější více pro běžné jazyky jako je C, než pro *Mathematicu*. Je však stále důležitý. Základ tohoto stylu programování je založen na cyklech (**`Do`**, **`While`**, **`For`**) a podmínětných příkazech (**`If`**, **`Which`**). V kapitole 5 knihy *An Introduction to Programming with Mathematica* [15] je například procedurální programování v programu *Mathematica* diskutováno během řešení klasických úloh jako je Newtonova metoda (neboli metoda tečen) a Eratosthenovo síto.

3. 3. 2 Funkcionální programování

Tento styl programování je záležitostí uživatelem vytvořených funkcí, které chápou výpočet jako vyhodnocení matematických funkcí. V podstatě tyto funkce mohou pracovat s libovolným výrazem (*expression*), včetně obrázků, textových řetězců či jiných funkcí. Tento styl programování odlišuje program *Mathematica* od jiných běžně používaných jazyků jako jsou Pascal, C++, Java a jiné. Výsledný kód bývá často kratší

a spolehlivější, než odpovídající kód zapsaný stylem procedurálního programování.

```
In[13]:= Tally[Characters["funkcionální vs. procedurální programování"]]
Out[13]= {{f, 1}, {u, 2}, {n, 5}, {k, 1}, {c, 2}, {i, 1}, {o, 4},
          {á, 3}, {l, 2}, {í, 3}, {  , 3}, {v, 2}, {s, 1}, {., 1},
          {p, 2}, {r, 4}, {e, 1}, {d, 1}, {g, 1}, {a, 1}, {m, 1}}

In[14]:= string = "funkcionální vs. procedurální programování";
          chars = Characters[string];
          Tally[chars]
Out[16]= {{f, 1}, {u, 2}, {n, 5}, {k, 1}, {c, 2}, {i, 1}, {o, 4},
          {á, 3}, {l, 2}, {í, 3}, {  , 3}, {v, 2}, {s, 1}, {., 1},
          {p, 2}, {r, 4}, {e, 1}, {d, 1}, {g, 1}, {a, 1}, {m, 1}}
```

Funkce můžeme dělit do několika kategorií:

- **Funkce pro manipulaci s výrazy**

  Na těchto funkcích je založeno funkcionální programování v *Mathematice*. Mezi nejdůležitější patří **Map**, **Thread** a **Apply**. Dále pak **MapThread**, **Inner** a **Outher**. **Map** aplikuje libovolnou funkci na každý element v *listu*. Stejně jako u několika běžných funkcí v *Mathematice*, je možné psát **Map** několika způsoby. První je klasická forma zápisu funkce **Map[f,*expr*]** a druhá je zkrácená notace **/@.** Použití **Map** lze demonstrovat na jednoduchém příkladě [15, str. 78].

```
In[17]:= Map[g, {a, b, c, d}];
In[18]:= g /@ {a, b, c, d}
Out[18]= {g[a], g[b], g[c], g[d]}
```

Funkce Thread mění operace za argumenty, které mají strukturu *listu*. Pro lepší pochopení uvádíme názornou ukázku [15, str. 79].

```
In[19]:= Thread[g[{1, 2, 3, 4}, {a, b, c, d}]]
Out[19]= {g[1, a], g[2, b], g[3, c], g[4, d]}
```

Oba *listy* v argumentu musí mít stejnou délku. Kdybychom vynechali funkci **g**, **Thread** by z uvedených listů vytvořil dvojice prvků stejných pozic.

Funkce Apply se používá ke změně struktury *výrazu*.

```
In[20]:= Apply [a, b[α, β, γ, δ, ϵ]]
Out[20]= a[α, β, γ, δ, ϵ]
```

Symbol **a** v tomto případě představuje tzv. hlavu (*Head*) *výrazu* $[\alpha,\beta,\gamma,\delta,\epsilon]$ [15, str. 82].

- **Iterační funkce**

  Používají se pro opakované použití funkce na nějaký *výraz* (*expression*).

Např.: **NestList**, **Nest**, **FoldList, Fold**.

In[21]:=
```
NestList[1/(1 + #) &, x, 3]
```
Out[21]=
$$\left\{x, \frac{1}{1+x}, \frac{1}{1+\frac{1}{1+x}}, \frac{1}{1+\frac{1}{1+\frac{1}{1+x}}}\right\}$$

In[22]:=
```
Nest[1/(1 + #) &, x, 3]
```
Out[22]=
$$\frac{1}{1+\frac{1}{1+\frac{1}{1+x}}}$$

In[23]:=
```
FoldList[#1^#2 &, x, {a, b, c, d}]
```
Out[23]=
$$\left\{x, x^a, (x^a)^b, \left((x^a)^b\right)^c, \left(\left((x^a)^b\right)^c\right)^d\right\}$$

In[24]:=
```
Fold[#1^#2 &, x, {a, b, c, d}]
```
Out[24]=
$$\left(\left((x^a)^b\right)^c\right)^d$$

- **Uživatelem definované funkce**

Používají se tehdy, kdy je potřeba řešit nějaké specifické případy, pro které nejsou v programu *Mathematica* definované funkce. Nebo pro potřebu uživatele upravit integrované funkce, tzv. *built-in functions.* Nově vytvořená funkce má strukturu podobnou matematické rovnici.

---

name[**arg**$_1$ _, **arg**$_2$ _, ..., **arg**$_n$ _] := **body**.

Levá strana je složená z názvu a série symbolů uzavřených v hranatých závorkách. Každý symbol zakončený podtržítkem (*blank*) vyjadřuje nějaký argument nově definované funkce. Podtržítko má význam objektového vzoru, který může vyjadřovat jakýkoliv *výraz*.

Pravá strana funkce se nazývá tělo (*body*), které může být buď jednoduchý *výraz* (*a one-liner*) nebo posloupnost *výrazů* (*a compound function*). Argumenty definované na levé straně funkce se vyskytují na pravé straně bez podtržítek. Jednoduše lze říct, že pravá strana je předpis, podle kterého probíhají výpočty s dosazenými hodnotami argumentu.

Pravá a levá strana jsou od sebe odděleny symbolem pro opožděné (*delayed*) přiřazení.

In[25]:=
```
f[x_] := x (1 - x^2)/3
```

In[26]:=
```
f[5]
```
Out[26]=
$$-40$$

- **Pomocné funkce**

- **Složené funkce**

  Složené funkce mají levou stranu definovanou obdobně, jako je tomu u funkcí definovaných uživatelem, kterou jsme popsali výše. Pravá strana je složená z po sobě jdoucích výrazů uzavřených v závorkách a odělených středníkem.

```
name[arg1_, arg2_, ..., argn_] := (expr1; expr2; ...; exprm).
```

Vyhodnocení funkce s konkrétními hodnotami argumentu je realizováno postupným vyhodnocováním výrazů $expr_i$ v daném pořadí. Jako výsledek se zobrazí výstup posledního výrazu $expr_m$ [15, str. 96].

- **Funkce `Module`, `Block` a `With`**

  Fce **`Block`** slouží k lokalizaci proměnných a **`Modul`** k lokalizaci názvů proměnných. Funkce **`With`** lokalizuje konstanty.

- **Pure Functions (“čisté funkce”)**

  Funkce *Pure Function* má dvě formy zápisu. První je klasická struktura **`Function[x, body]`**a druhá je zkrácená notace reprezentovaná znaky **#** a **&**. Mřížka **#** má význam proměnné a ampersand **&** nám říká, že jde právě o *Pure Function.*

```
In[27]:= Function[x, 1 + 3^x][2]
Out[27]= 10

In[28]:= (1 + 3^# &)[2]
Out[28]= 10
```

Má uplatnění především v případech, kdy ji použijeme jen jednou, a to jako argument funkce vyššího řádu, např. : **`Folder`**, **`Nest`** nebo **`Map`** [15, str. 102].

3. 3. 3 Programování rule-based

Programování rule-based, jak už název napovídá, je styl programování založený na pravidlech. Uplatňuje se především v kombinaci s jinými běžnými programovacími styly (procedurální, funkcionální). Pravidla můžou například měnit formu *výrazu*, různě modifikovat jeho části podle nějakého vrozu (*pattern*) nebo jednoduše separovat jednotlivé elementy *výrazu* pomocí transformačních pravidel (*transformation rules*).

Základní strukturou rule-based programování je tzv. *blank*, neboli podtržítko. Jak již bylo zmíněno v sekci funkcionálního programování, podtržítko má význam *vzoru*. Používají se tři formy, jednoduché (_, *single*), dvojté (__, *double*) a trojté (___, *triple*). Dvojté podtržítko se označuje jako *BlankSequence* a trojté jako *BlankNullSequence*. Nejlépe lze jejich význam ukázat na příkladech.

```
In[29]:= Cases[{{a, b}, {}, {2^(x+y)}, {1, 0}, {c, d, e}, {⌘, ⌃}, {"Blank"}}, {p_}]
Out[29]= {{2^(x+y)}, {Blank}}
```

Jednoduché podtržítko představuje *výraz*, který má pouze jeden symbol.

```
In[30]:= Cases[{{a, b}, {}, {2^(x+y)}, {1, 0},
           {c, d, e}, {⌘, ⌃}, {"BlankSequence"}}, {p__}]
Out[30]= {{a, b}, {2^(x+y)}, {1, 0}, {c, d, e}, {⌘, ⌃}, {BlankSequence}}
```

Dvojté podtržítko představuje *vzor*, podle kterého se vypisují *výrazy* s jedním nebo více symboly.

```
In[31]:= Cases[{{a, b}, {}, {2^(x+y)}, {1, 0},
           {c, d, e}, {⌘, ⌃}, {"BlankNullSequence"}}, {p___}]
Out[31]= {{a, b}, {}, {2^(x+y)}, {1, 0}, {c, d, e}, {⌘, ⌃}, {BlankNullSequence}}
```

Vzor trojtého podtržítka reprezentuje *výrazy* s jakýmkoliv počtem symbolů, tedy i *výraz*, který neobsahuje žádný symbol.

V některých případech je potřeba upřesnit typ výrazu, který má podléhat vzoru. Toho docílíme připsáním dané specifikace za podtržítko (viz. tab. 3).

VZOR	TYP *VÝRAZU*
x_	jakýkoliv *výraz*
x_Integer	celé číslo
x_List	*list*
x_Real	reálné číslo
x_Symbol	symbol

Tab. 3: Vzor a jemu odpovídající typ proměnné.

```
In[32]:= Cases[{1, -1, 3.5, π, -0.128, 1/3, ♡}, _Integer]
Out[32]= {1, -1}

In[33]:= Cases[{1, -1, 3.5, π, -0.128, 1/3, ♡}, _Symbol]
Out[33]= {π, ♡}
```

Dále můžeme pomocí zkrácené notace `/;` klást na *vzory* podmínky.

```
In[34]:= ff[x_Real /; x > 0] := x + Log[x]

In[35]:= {ff[15.], ff[-3.5], ff[0.], ff[0.987], ff[π]}
Out[35]= {17.7081, ff[-3.5], ff[0.], 0.973915, ff[π]}
```

Transformační pravidla se vyskytují v programu *Mathematica* běžně. Jsou používána k zobrazení řešení rovnic, využívají se jako prostředek ke specifikaci možností (*options*) funkcí a také tvoří základ většiny algebraických manupulací. Princip použití těchto pravidel budeme demonstrovat pomocí pravidel přiřazení (*assignment*) a přemístění (*replacement*).

Pravidlo přiřazení se běžně používá u definování nových funkcí nebo přiřazení jednoho *výrazu výrazu* jinému.

```
In[36]:= soucet[x_, y_, z_] := x + y + z
         soucet[a, b, c]
Out[37]= a + b + c

In[38]:= k = l + m;
         k + l + m
Out[39]= 2 l + 2 m
```

V případě pravidla přemístění, stejně jako u přiřazení, rozlišujeme dvě formy, a to okamžité (*immediate*) a opožděné (*delayed*). V případě okamžitého pravidla je přemístění vyhodnoceno okamžitě. Ve standardní podobě se zapisuje jako funkce **Rule[***vzor, přemístění***]** a jeho zkrácená notace je vyjádřená šipkou, *vzor* → *přemístění*.

```
In[40]:= {a, a} /. Rule[a, Random[]]
Out[40]= {0.418154, 0.418154}

In[41]:= {a, a} /. a → Random[]
Out[41]= {0.809167, 0.809167}
```

Symbol **/.** má význam substituce. Je to zkrácená notace funkce **ReplaceAll[***výraz, pravidlo***]**, která aplikuje pravidlo pro přemístění na *výraz*.

Opožděné přemístění má standardní formu ve tvaru **RuleDelayed[***vzor, přemístění***]** a zkrácená notace se píše pomocí symbolu :→.

```
In[42]:= {a, a} /. RuleDelayed[a, Random[]]
Out[42]= {0.827575, 0.367602}

In[43]:= {a, a} /. a :→ Random[]
Out[43]= {0.923171, 0.501927}
```

Jak lze vidět z příkladu, u opožděného přiřazení nejdříve proběhne přiřazení a následně pak vyhodnocení. Naopak u okamžitého přiřazení je funkce **Random[]** nejdříve vyhodnocena a výsledek je přiřazen každému **a**. Ve výsledku jsme tedy obdrželi *list* dvou stejných náhodných čísel.

Podobnou substituční vlastnost jako má funkce **ReplaceAll**, má také funkce **ReplaceRepeated**. Rozdíl mezi těmito dvěma funkcemi spočívá v tom, že **ReplaceAll** substituce proběhne pouze u prvního odpovídajícího výskytu. U **ReplaceRepeated** je substituce opakovaně aplikovaná, dokud neproběhnout veškeré změny [15].

```
In[44]:= a b c d /. x_ y_ → x + y
Out[44]= a + b c d
```

In[45]:=
```
a b c d //. x_ y_ → x + y
```

Out[45]=
$$a + b + c + d$$

4. Vlastní zpracování

Zpracování dat provedeme ve dvou fázích, nejdříve zpracujeme soubor dat jednoho měření a pak postup zobecníme pro zbývající.

4. 1 Postup zpracování jednoho měření

Ze všeho nejdříve si zjistíme, kde je právě používaný *notebook* umístěný. K tomu slouží funkce **NotebookDirectory**. Tuto cestu následně nastavíme příkazem **SetDirectory** jako výchozí pro funkce **Import** a **Export**.

In[46]:=
```
SetDirectory[NotebookDirectory[]]
```

Out[46]=
```
F:\Zdrojova_data_desy_2013
```

Data z libovolného souboru, který *Mathematica* podporuje, nahrajeme do *notebooku* funkcí **Import**. Ta má dva argumenty. Umístění souboru s jeho názvem a formát, ve kterém se data zobrazí.

In[47]:=
```
importteploty = Import["DESY\\Hempsod05_320.txt", "Table"]
```

Out[47]=

A very large output was generated. Here is a sample of it:

```
{{date/time, sec., from, start, Temp, 1, Temp, 2, SETpoint},
            {6.10.2013, 4:49:51, 0, 0., 25.1, 0},
            {6.10.2013, 4:49:52, 2, 0., 25.3, 0},
            {6.10.2013, 4:49:54, 3, 0., 25.3, 0},
            {6.10.2013, 4:49:56, 5, 0., 25.3, 0},
      {6.10.2013, 4:49:57, 7, 0., 25.3, 0}, <<1278>>,
        {6.10.2013, 5:31:25, 2494, 0., 329.8, 330},
        {6.10.2013, 5:31:27, 2496, 0., 329.9, 330},
        {6.10.2013, 5:31:29, 2498, 0., 329.8, 330},
        {6.10.2013, 5:31:31, 2500, 0., 329.9, 330},
        {6.10.2013, 5:31:33, 2502, 0., 329.9, 330},
        {6.10.2013, 5:32:41, 2570, 0., 329.9, 330}}
```

Show Less | Show More | Show Full Output | Set Size Limit...

V našem případě jsme použili formát **Table.** Je to příkaz, pomocí kterého se data zobrazují jako *list*, jehož prvky jsou také *listy* reprezentující řádky souboru.

Jak je patrné z výstupu `Out[47]` , načtená data obsahují i hlavičku tabulky. Tu odstraníme použitím funkce **DeleteCases**. Argumenty jsou *list* a *výraz*, který má být z *listu* odstraněn.

In[48]:=
```
deleteimport = DeleteCases[importteploty, {"date/time",
"sec.", "from", "start", "Temp", 1, "Temp", 2, "SETpoint"}]
```

Out[48]=

A very large output was generated. Here is a sample of it:

```
{{6.10.2013, 4:49:51, 0, 0., 25.1, 0},
 {6.10.2013, 4:49:52, 2, 0., 25.3, 0},
 {6.10.2013, 4:49:54, 3, 0., 25.3, 0},
 {6.10.2013, 4:49:56, 5, 0., 25.3, 0},
 {6.10.2013, 4:49:57, 7, 0., 25.3, 0},
 {6.10.2013, 4:49:59, 8, 0., 25.3, 0},
{6.10.2013, 4:50:01, 10, 0., 25.3, 0}, <<1276>>,
  {6.10.2013, 5:31:25, 2494, 0., 329.8, 330},
  {6.10.2013, 5:31:27, 2496, 0., 329.9, 330},
  {6.10.2013, 5:31:29, 2498, 0., 329.8, 330},
  {6.10.2013, 5:31:31, 2500, 0., 329.9, 330},
  {6.10.2013, 5:31:33, 2502, 0., 329.9, 330},
  {6.10.2013, 5:32:41, 2570, 0., 329.9, 330}}
```

Show Less | Show More | Show Full Output | Set Size Limit...

Dále separujeme z *listu* jen potřebné informace, tedy čas a příslušnou teplotu. K tomu použijeme funkci **ReplaceRepeated**.

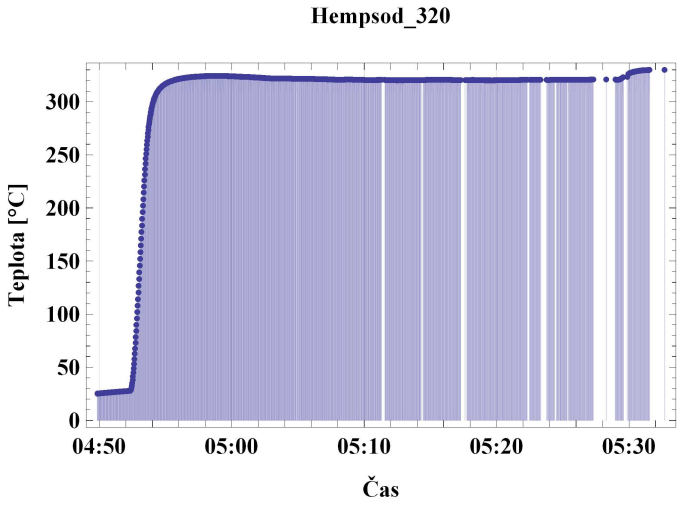
In[49]:=
```
ReplaceRepeated[deleteimport, {a_, b_, c_, d_, e_, f_} → {b, e}];
```

In[50]:=
```
teploty = deleteimport //. {a_, b_, c_, d_, e_, f_} → {b, e};
```

Pomocí funkce **DateListPlot** vyneseme do grafu závislost teploty na čase (obr. 3). Tato funkce patří do kategorie funkcí **ListPlot** zobrazující data vektorového nebo maticového formátu jako body v grafu [15, str. 282].

In[51]:=
```
DateListPlot[teploty, ImageSize → 350, AxesOrigin → {Automatic, 0},
 Filling → Bottom, LabelStyle → Directive[Bold, Medium],
 PlotLabel → "Hempsod_320", FrameLabel → {"Čas", "Teplota [°C]"}]
```

Out[51]=



Obr. 3: Graf závislosti teploty na čase.

Abychom mohli s daty lépe manipulovat, převedeme datum a čas na tzv. absolutní čas. Toho docílíme použitím fukce **AbsoluteTime**. Protože jednou z možných

forem argumentu této funkce je textový řetězec (*string*), změníme formát data a času z obecného *výrazu* (*expression*) na *string*. Využijeme k tomu funkci **ToString**. Před tím musíme také změnit zápis data. V textovém souboru je datum ve formátu 6.10.2013 a v souborech *.fio ve formátu 6-Oct-2013. Abychom měli jednotné formáty a vyhnuli se možným komplikacím, změníme datum textového souboru na formát, ve kterém je datum v souborech *.fio. Použijeme k tomu funkci **ReplaceAll** v její zkrácené notaci **/.**.

In[52]:=
```
timeformat = deleteimport //. {a_, b_, c_, d_, e_, f_} → {a , b} /.
                "6.10.2013" → "6-Oct-2013";
```

In[53]:=
```
absolutetime = Map[AbsoluteTime[#] &, Map[ToString[#] &, timeformat]];
```

Pro další zpracování bude výhodné vytvořit *list* dvojic absolutních časů a jím odpovídajících teplot. Existuje více funkcí, které disponují požadovanou vlastností, např.: **Transpose**, **Thread**, **Join**. Je tu i taková možnost, že si potřebnou funkci sami nadefinujeme. Pro jednoduchost zvolíme funkci **Thread**.

In[54]:=
```
data = Thread[
 {absolutetime, deleteimport //. {a_, b_, c_, d_, e_, f_} → e}];
```

List **data** budeme třídit na intervaly. Krajními body budou časy začátku a konce jednotlivých měření spekter NFS.

V dalším kroku načteme data ze všech souborů *.fio, které přísluší textovému souboru *Hempsod05_320*. V případě textového souboru jsme importovali pouze jeden soubor, nyní však budeme importovat více souborů současně. Použijeme funkci **Import**, tentokrát v kombinaci s **Map**, **FileNames** a *Pure Function*. **FileNames** má jako argument umístění s obecně daným názvem souboru s příponou. Výstupem je *list* *.fio souborů, které jsou na explicitně určeném místě v počítači.

In[55]:=
```
timeimport = Map[Import[#, "Table"] &,
 FileNames["2013oct02\\sort\\HemPSod\\HemPSod05_320\\*.fio"]]
```

Out[55]=

A very large output was generated. Here is a sample of it:

{ <<1>> }

Show Less | Show More | Show Full Output | Set Size Limit...

Načtené soubory obsahují kromě času zápisu měření i další informace, které jsou pro tuto práci nepodstatné. K separaci časů umístěných v 6. řádku a 3. sloupci jsou vhodné funkce **Part**, **Map** a také *Pure Function.* Prvním argumentem funkce **Part** je *list* (obecně však libovolný *výraz*) a druhý argument je pozice části *výrazu*, kterou tato funkce vypíše. Její zkrácená notace je dána dvojtými hranatými závorkami **[[ ]]**.

In[56]:=
```
time = Map[Part[#, 6, 3] &, timeimport]
```

Out[56]=
```
{6-Oct-2013,04:51:04, 6-Oct-2013,04:52:15, 6-Oct-2013,04:53:25,
 6-Oct-2013,04:54:35, 6-Oct-2013,04:55:45, 6-Oct-2013,04:56:55,
 6-Oct-2013,04:59:34, 6-Oct-2013,05:00:45, 6-Oct-2013,05:01:55,
 6-Oct-2013,05:03:05, 6-Oct-2013,05:04:15, 6-Oct-2013,05:05:26,
 6-Oct-2013,05:06:36, 6-Oct-2013,05:07:46, 6-Oct-2013,05:08:57,
 6-Oct-2013,05:10:08, 6-Oct-2013,05:11:18, 6-Oct-2013,05:12:28,
 6-Oct-2013,05:13:39, 6-Oct-2013,05:14:49, 6-Oct-2013,05:15:59,
 6-Oct-2013,05:17:09, 6-Oct-2013,05:18:20, 6-Oct-2013,05:19:31,
 6-Oct-2013,05:20:41, 6-Oct-2013,05:21:52, 6-Oct-2013,05:23:02,
 6-Oct-2013,05:24:12, 6-Oct-2013,05:25:22, 6-Oct-2013,05:26:33,
 6-Oct-2013,05:27:43, 6-Oct-2013,05:28:53, 6-Oct-2013,05:30:04,
     6-Oct-2013,05:31:14, 6-Oct-2013,05:32:24, 6-Oct-2013}
```

Obdrželi jsme *list* s daty a časy, které převedeme funkcí **AbsoluteTime** opět na počet sekund od roku 1900.

In[57]:=
```
finitetime = Map[AbsoluteTime[#] &, time]
```

Out[57]=
```
{3 590 023 864, 3 590 023 935, 3 590 024 005, 3 590 024 075,
 3 590 024 145, 3 590 024 215, 3 590 024 374, 3 590 024 445,
 3 590 024 515, 3 590 024 585, 3 590 024 655, 3 590 024 726,
 3 590 024 796, 3 590 024 866, 3 590 024 937, 3 590 025 008,
3 590 025 078, 3 590 025 148, 3 590 025 219, 3 590 025 289, 3 590 025 359,
3 590 025 429, 3 590 025 500, 3 590 025 571, 3 590 025 641, 3 590 025 712,
3 590 025 782, 3 590 025 852, 3 590 025 922, 3 590 025 993, 3 590 026 063,
3 590 026 133, 3 590 026 204, 3 590 026 274, 3 590 026 344, 3 590 006 400}
```

Odečtením 60 s od každého prvku v *listu* **finitetime** získáme počáteční časy měření. Použijeme podobnou konstrukci příkazu jako pro **finitetime**.

In[58]:=
```
initialtime = Map[# - 60 &, finitetime]
```

Out[58]=
```
{3 590 023 804, 3 590 023 875, 3 590 023 945, 3 590 024 015,
 3 590 024 085, 3 590 024 155, 3 590 024 314, 3 590 024 385,
 3 590 024 455, 3 590 024 525, 3 590 024 595, 3 590 024 666,
 3 590 024 736, 3 590 024 806, 3 590 024 877, 3 590 024 948,
3 590 025 018, 3 590 025 088, 3 590 025 159, 3 590 025 229, 3 590 025 299,
3 590 025 369, 3 590 025 440, 3 590 025 511, 3 590 025 581, 3 590 025 652,
3 590 025 722, 3 590 025 792, 3 590 025 862, 3 590 025 933, 3 590 026 003,
3 590 026 073, 3 590 026 144, 3 590 026 214, 3 590 026 284, 3 590 006 340}
```

Listy **initialtime** a **finitetime** zkompletujeme funkcí **Thread** do jednoho *listu*. Tím nám vznikne *list* dvojic počátečních a koncových časů.

In[59]:=
```
timedata = Thread[{initialtime, finitetime}];
```

Rozdělení *listu* **data** do intervalů, které jsou dány počátečními a koncovými časy měření, provedeme porovnáním. Nejdříve si postup ukážeme pro první interval a následně pak pro všechny.

Podmínka, kterou použijeme pro vybrání časů, které spadají mezi počáteční a koncový čas, je vytvořená pomocí zkrácených notací funkcí **Part** (**[[]]**) a **And** (**&&**). **And** je funkce logické spojky *a*, nebo-li konjunkce. Výstupem je pravdivostní ohodno-

cení jejího argumentu (**True** nebo **False**). Celá konstrukce podmínky je vložená do funkce **Table** jako její argument. **Table** se používá pro tvorbu *listu*. Jejími argumenty jsou *výraz* (*expression*) a *iterator*. *Iterator* je *list* ve tvaru **{i, i_{MIN}, i_{MAX}}**, který udává, kolikrát bude *výraz* v prvním argumentu vyhodnocen a také počet prvků ve vytvořeném *listu* [15, str. 56].

In[60]:=
```
Table[((timedata[[1, 1]] ≤ data[[i, 1]]) &&
    (data[[i, 1]] ≤ timedata[[1, 2]])),
  {i, 1, Length[data]}] // Take[#, 50] &
```

Out[60]=
```
{False, False, False, False, False, False, False, False, True,
 True, True, True, True, True, True, True, True, True, True,
  True, True, True, True, True, True, True, True, True, True,
  True, True, True, True, True, True, True, True, True, True,
 True, True, True, True, True, True, False, False, False, False}
```

Výstupem je *list* s **True** a **False**. Pro nás jsou podstatné jen ty prvky, které podmínku splnili a tudíž mají hodnotu **True**. Funkcí **Position** zjistíme jejich pozice.

In[61]:=
```
pozice1 = Flatten[Position[Table[((timedata[[1, 1]] ≤ data[[i, 1]]) &&
    (data[[i, 1]] ≤ timedata[[1, 2]])), {i, 1, Length[data]}], True]]
```

Out[61]=
```
{9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27,
 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46}
```

Listy **data** a **pozice1** dosadíme do funkce **Part**. Výstup pak vložíme do funkce **Table**, tím získáme hledaný interval.

In[62]:=
```
Interval1 = Table[Part[data, pozice1[[k]]],
  {k, 1, Length[pozice1]}] // Take[#, 15] &
```

Out[62]=
```
{{3 590 023 804, 25.4}, {3 590 023 805, 25.4}, {3 590 023 807, 25.5},
 {3 590 023 809, 25.5}, {3 590 023 810, 25.5}, {3 590 023 812, 25.5},
 {3 590 023 813, 25.6}, {3 590 023 815, 25.6}, {3 590 023 817, 25.7},
 {3 590 023 818, 25.7}, {3 590 023 820, 25.7}, {3 590 023 822, 25.8},
 {3 590 023 823, 25.8}, {3 590 023 825, 25.8}, {3 590 023 826, 25.8}}
```

Postup, který jsme popsali pro první interval, nyní zobecníme pro všechny intervaly definované v *listu* **timedata**. To znamená, že zaměníme následující výrazy

timedata[[1,1]] ↦ timedata[[j,1]],

timedata[[1,2]] ↦ timedata[[j,2]],

pozice1[[k]] ↦ pozice[[k,r]]

a celé příkazy vytvořené pro zpracování jednoho intervalu vložíme do funkce **Table**, jako její argument.

In[63]:=
```
pozice =
 Table[Flatten[Position[Table[((timedata[[j, 1]] ≤ data[[i, 1]]) &&
     (data[[i, 1]] ≤ timedata[[j, 2]])), {i, 1, Length[data]}],
   True]], {j, 1, Length[timedata]}] // DeleteCases[#, {}] &;
```

Zápis *výraz*//**DeleteCases[#,***vzor***]&** je obdobou námi používaného zápisu **DeleteCases[***výraz***,***vzor***]**.

In[64]:=
```
intervaly = Table[Part[data, pozice[[k, l]]],
  {k, 1, Length[pozice]}, {l, 1, Length[pozice[[k]]]}];
```

Pro zajímavost můžeme zjistit celkový počet počet intervalů pompocí příkazu **Lenght**.

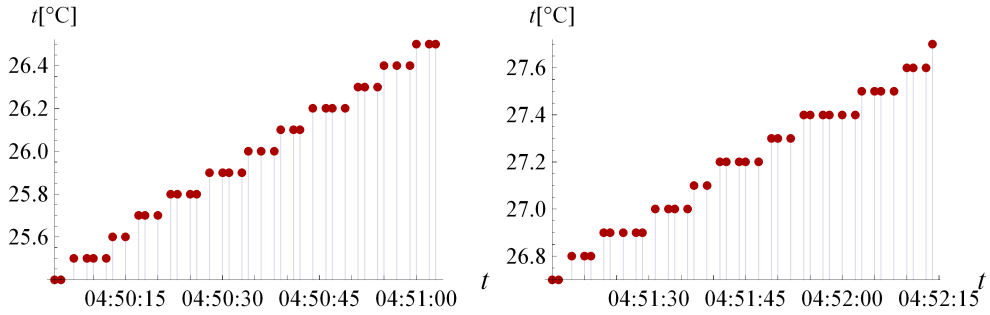
In[65]:=
```
Length[intervaly]
```

Out[65]=
```
35
```

Tohoto jsme využili v *iteratoru* fce **Table**. **Length** se může použít pro zpracování více datových souborů, které mají obecně různý počet dat.

Ke zpracovávání dat patří neodmyslitelně také jejich vizualizace, tedy znázornění těchto dat v grafu. Vzhledem k tomu, že máme *list* intervalů, kde každý interval má být vynesem do grafu, použijeme funkci **DateListPlot** v kombinaci s funkcemi **Map** a *Pure Function* (**# &**).

In[66]:=
```
grafy = Map[DateListPlot[#, Axes → True, Frame → False,
    GridLines → None, Filling → Axis, ImageSize → 250,
  PlotStyle → Directive[PointSize[Medium], Darker[Red]],
    AxesLabel → {Style["   t", 15], Style["t[°C]", 12]},
    LabelStyle → Directive[Black, 12]] &, intervaly];
```



Obr. 5: Ukázka grafů časové závislosti teplot prvních dvou intervalů souboru Hempsod05_320.txt.

Abychom se dokázali orientovat ve velkém množství dat, která jsou již rozdělena do intervalů, sestavíme přehledovou tabulku. Každý její řádek bude jednotlivé intervaly charakterizovat minimální, maximální a průměrnou teplotou, názvem a číslem příslušejícího souboru. Dále pak datem a počátečním i koncovým časem měření spekter.

Nejdříve vypíšeme funkcí **ReplaceRepeated** z intervalů pouze teploty.

In[67]:=
```
temp = Table[ReplaceRepeated[intervaly[[m]], {a_, b_} → b],
    {m, 1, Length[intervaly]}];
```

Dále z každého intervalu vybereme minimální, maximální a průměrnou teplotu. V programu *Mathematica* jsou přímo k tomu definované funkce **Min**, **Max** a **Mean**. Argument těchto funkcí je *list* čísel, která následně vyhodnotí dle svého zabudovaného algoritmu. Protože budeme zpracovávat všechny intervaly sooučasně, budeme funkce kombinovat s **Table**, **Length** a **Map** spolu s *Pure Function*. Upravíme také formát čísel

příkazem **NumberForm**, jehož argumenty jsou číslo a možnosti jeho zobrazení. První prvek ve složené závorce druhého argumentu udává počet platných číslic a druhý prvek počet desetinných míst vpravo od desetinné čárky.

```
In[68]:= min = Map[NumberForm[#, {4, 1}] &,
            Table[Min[temp[[n]]], {n, 1, Length[intervaly]}]]
Out[68]= {25.4, 26.7, 30.8, 259.4, 314.7, 321.2, 324.1, 323.4, 322.8,
           321.8, 321.7, 321.5, 321.3, 320.7, 320.6, 320.4, 320.3,
         320.1, 320.1, 320.2, 320.6, 320.3, 320.3, 320.1, 320.1, 320.2,
         320.5, 320.4, 320.6, 320.6, 320.7, 320.8, 320.5, 327.8, 329.8}

In[69]:= max = Map[NumberForm[#, {4, 1}] &,
            Table[Max[temp[[n]]], {n, 1, Length[intervaly]}]]
Out[69]= {26.5, 27.7, 225.8, 312.8, 320.6, 323.2, 324.3, 324.0, 323.4,
           322.6, 321.9, 321.7, 321.5, 321.2, 320.9, 320.7, 320.5,
         320.5, 320.3, 320.5, 320.7, 320.5, 320.5, 320.3, 320.3, 320.5,
         320.6, 320.6, 320.7, 320.7, 320.8, 320.8, 326.9, 329.7, 329.9}

In[70]:= mean = Map[NumberForm[#, {4, 1}] &,
            Table[Mean[temp[[n]]], {n, 1, Length[intervaly]}]]
Out[70]= {26.0, 27.2, 118.5, 295.9, 318.2, 322.3, 324.2, 323.7, 323.1,
           322.2, 321.8, 321.5, 321.4, 321.0, 320.7, 320.5, 320.4,
         320.3, 320.2, 320.3, 320.6, 320.4, 320.4, 320.2, 320.2, 320.4,
         320.6, 320.5, 320.7, 320.7, 320.7, 320.8, 322.5, 329.0, 329.9}
```

Jako výstup jsme dostali *list* požadovaných hodnot, které jsou vybrané z jednotlivých intervalů.

Nyní si načteme *list* počátečních časů a *list* koncových časů ve formátu HH:MM:SS a *list* dat ve formátu D -“zkratka měsíce”- RRRR. Využijeme k tomu funkce **Map** s *Pure Function* a **DateString**. **DateString**, jak už její název napovídá, převede absolutní čas na námi určený formát ať už času nebo data.

```
In[71]:= initial = Map[DateString[#, {"Hour", ":", "Minute", ":", "Second"}] &,
            Table[timedata[[o, 1]], {o, 1, Length[timedata]}]]
Out[71]= {04:50:04, 04:51:15, 04:52:25, 04:53:35, 04:54:45, 04:55:55,
          04:58:34, 04:59:45, 05:00:55, 05:02:05, 05:03:15, 05:04:26,
          05:05:36, 05:06:46, 05:07:57, 05:09:08, 05:10:18, 05:11:28,
          05:12:39, 05:13:49, 05:14:59, 05:16:09, 05:17:20, 05:18:31,
          05:19:41, 05:20:52, 05:22:02, 05:23:12, 05:24:22, 05:25:33,
          05:26:43, 05:27:53, 05:29:04, 05:30:14, 05:31:24, 23:59:00}

In[72]:= finite = Map[DateString[#, {"Hour", ":", "Minute", ":", "Second"}] &,
            Table[timedata[[o, 2]], {o, 1, Length[timedata]}]]
Out[72]= {04:51:04, 04:52:15, 04:53:25, 04:54:35, 04:55:45, 04:56:55,
          04:59:34, 05:00:45, 05:01:55, 05:03:05, 05:04:15, 05:05:26,
          05:06:36, 05:07:46, 05:08:57, 05:10:08, 05:11:18, 05:12:28,
          05:13:39, 05:14:49, 05:15:59, 05:17:09, 05:18:20, 05:19:31,
          05:20:41, 05:21:52, 05:23:02, 05:24:12, 05:25:22, 05:26:33,
          05:27:43, 05:28:53, 05:30:04, 05:31:14, 05:32:24, 00:00:00}
```

In[73]:=
```
datum =
  Map[DateString[#, {"Day", "-", "MonthNameShort", "-", "Year"}] &,
    Table[timedata[[o, 1]], {o, 1, Length[timedata]}]]
```

Out[73]=
```
{06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013,
 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013,
 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013,
 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013,
 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013,
 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013,
 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013,
 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 06-Oct-2013, 05-Oct-2013}
```

Název a číslo souboru budeme separovat funkcemi **StringDrop** a **StringTake** z cesty k souborům *.fio. Funkce **StringDrop[***string***,-n]** vezme posledních *n* znaků a vynechá je. Naopak funkce **StringTake[***string***,-n]** vypíše pouze posledních *n* znaků. V obou případech bude jejich argumentem výstup složené funkce z funkcí **FileNames**, **FileBasedName**, **Map** v kombinaci s *Pure Functio*n a **Flatten**.

In[74]:=
```
nazevsouboru = StringDrop[Flatten[Map[FileBaseName, FileNames[
    "2013oct02\\sort\\HemPSod\\HemPSod05_320\\*.fio"]]], -6]
```

Out[74]=
```
{hempsod05_320, hempsod05_320, hempsod05_320, hempsod05_320,
 hempsod05_320, hempsod05_320, hempsod05_320, hempsod05_320,
 hempsod05_320, hempsod05_320, hempsod05_320, hempsod05_320,
 hempsod05_320, hempsod05_320, hempsod05_320, hempsod05_320,
 hempsod05_320, hempsod05_320, hempsod05_320, hempsod05_320,
 hempsod05_320, hempsod05_320, hempsod05_320, hempsod05_320,
 hempsod05_320, hempsod05_320, hempsod05_320, hempsod05_320,
 hempsod05_320, hempsod05_320, hempsod05_320, hempsod05_320_rt}
```

In[75]:=
```
cislasouboru = StringTake[Flatten[Map[FileBaseName, FileNames[
    "2013oct02\\sort\\HemPSod\\HemPSod05_320\\*.fio"]]], -5]
```

Out[75]=
```
{00001, 00002, 00003, 00004, 00005, 00006, 00007, 00008, 00009,
 00010, 00011, 00012, 00013, 00014, 00015, 00016, 00017, 00018,
 00019, 00020, 00021, 00022, 00023, 00024, 00025, 00026, 00027,
 00028, 00029, 00030, 00031, 00032, 00033, 00034, 00035, 00001}
```

Vytvořené *listy* **min**, **max**, **mean**, **initial**, **finite**, **datum**, **nazevsouboru** a **cislasouboru** zkompletujeme použitím funkcí **Table** a **Join**. Tímto vytvoříme *list*, jehož prvky budou opět *listy* s parametry jednotlivých intervalů. K tomuto připojíme funkcí **Prepend** *list*, který bude představovat hlavičku tabulky.

In[76]:=
```
vysledek = Prepend[
  Table[Join[{cislasouboru[[i]], min[[i]], max[[i]], mean[[i]],
    nazevsouboru[[i]], datum[[i]], initial[[i]], finite[[i]]}],
    {i, 1, Length[max]}], {"č.s.", "MIN[°C]", "MAX[°C]",
    "MEAN[°C]", "SOUBOR", "DATUM", "ZAČÁTEK", "KONEC"}];
```

Funkce **Grid** převede vnořené *listy listu* **vysledek** na řádky tabulky. Každý prvek vnořených *listů* tvoří sloupce tabulky.

In[77]:=
```
tabulka =
Grid[vysledek, Frame → All] // Style[#, FontFamily → "Arial", 10] &
```

Out[77]=

č.s.	MIN[℃]	MAX[℃]	MEAN[℃]	SOUBOR	DATUM	ZAČÁTEK	KONEC
00001	25.4	26.5	26.0	hempsod05_320	06–Oct–2013	04:50:04	04:51:04
00002	26.7	27.7	27.2	hempsod05_320	06–Oct–2013	04:51:15	04:52:15
00003	30.8	225.8	118.5	hempsod05_320	06–Oct–2013	04:52:25	04:53:25
00004	259.4	312.8	295.9	hempsod05_320	06–Oct–2013	04:53:35	04:54:35
00005	314.7	320.6	318.2	hempsod05_320	06–Oct–2013	04:54:45	04:55:45
00006	321.2	323.2	322.3	hempsod05_320	06–Oct–2013	04:55:55	04:56:55
00007	324.1	324.3	324.2	hempsod05_320	06–Oct–2013	04:58:34	04:59:34
00008	323.4	324.0	323.7	hempsod05_320	06–Oct–2013	04:59:45	05:00:45
00009	322.8	323.4	323.1	hempsod05_320	06–Oct–2013	05:00:55	05:01:55
00010	321.8	322.6	322.2	hempsod05_320	06–Oct–2013	05:02:05	05:03:05
00011	321.7	321.9	321.8	hempsod05_320	06–Oct–2013	05:03:15	05:04:15
00012	321.5	321.7	321.5	hempsod05_320	06–Oct–2013	05:04:26	05:05:26
00013	321.3	321.5	321.4	hempsod05_320	06–Oct–2013	05:05:36	05:06:36
00014	320.7	321.2	321.0	hempsod05_320	06–Oct–2013	05:06:46	05:07:46
00015	320.6	320.9	320.7	hempsod05_320	06–Oct–2013	05:07:57	05:08:57
00016	320.4	320.7	320.5	hempsod05_320	06–Oct–2013	05:09:08	05:10:08
00017	320.3	320.5	320.4	hempsod05_320	06–Oct–2013	05:10:18	05:11:18
00018	320.1	320.5	320.3	hempsod05_320	06–Oct–2013	05:11:28	05:12:28
00019	320.1	320.3	320.2	hempsod05_320	06–Oct–2013	05:12:39	05:13:39
00020	320.2	320.5	320.3	hempsod05_320	06–Oct–2013	05:13:49	05:14:49
00021	320.6	320.7	320.6	hempsod05_320	06–Oct–2013	05:14:59	05:15:59
00022	320.3	320.5	320.4	hempsod05_320	06–Oct–2013	05:16:09	05:17:09
00023	320.3	320.5	320.4	hempsod05_320	06–Oct–2013	05:17:20	05:18:20
00024	320.1	320.3	320.2	hempsod05_320	06–Oct–2013	05:18:31	05:19:31
00025	320.1	320.3	320.2	hempsod05_320	06–Oct–2013	05:19:41	05:20:41
00026	320.2	320.5	320.4	hempsod05_320	06–Oct–2013	05:20:52	05:21:52
00027	320.5	320.6	320.6	hempsod05_320	06–Oct–2013	05:22:02	05:23:02
00028	320.4	320.6	320.5	hempsod05_320	06–Oct–2013	05:23:12	05:24:12
00029	320.6	320.7	320.7	hempsod05_320	06–Oct–2013	05:24:22	05:25:22
00030	320.6	320.7	320.7	hempsod05_320	06–Oct–2013	05:25:33	05:26:33
00031	320.7	320.8	320.7	hempsod05_320	06–Oct–2013	05:26:43	05:27:43
00032	320.8	320.8	320.8	hempsod05_320	06–Oct–2013	05:27:53	05:28:53
00033	320.5	326.9	322.5	hempsod05_320	06–Oct–2013	05:29:04	05:30:04
00034	327.8	329.7	329.0	hempsod05_320	06–Oct–2013	05:30:14	05:31:14
00035	329.8	329.9	329.9	hempsod05_320	06–Oct–2013	05:31:24	05:32:24

Tab. 4: Tabulka charaktrictických hodnot jednotlivých intervalů souboru Hempsod05_320.txt.

Tabulku 4 exportujeme příkazem **Export** do textového souboru Hempsod05_320 s příponou .dat.

```
Export["Hempsod05_320.dat", vysledek, Alignment → Right]
```

Poslední částí zpracování dat je jejich interaktivní vizualizace v podobě výstupu funkce **Manipulate**. Forma výstupu zahrnuje:

- výběr souboru
- zobrazení příslušného grafu
- zobrazení odpovídající tabulky se základními informacemi
- možnost zobrazení souhrnné tabulky 4.

Struktura funkce **Manipulate** má následující tvar.

```
Manipulate[{{tabulka, grafy}, souhrnná tabulka}, možnosti manipulace]
```

Nejdříve odstraníme funkcí Drop z listu výsledek jeho první prvek, který obsahuje hlavičku tabulky. Takto upravený list použijeme jako argument funkce **ReplaceRepeated** a vybereme z něj pouze časy a teploty.

In[78]:=
```
elements = ReplaceRepeated[Drop[vysledek, 1],
  {a_, b_, c_, d_, e_, f_, g_, h_} → {b, c, d, g, h}];
```

Prvky *listu* **elements** uspořádáme funkcí **TableForm** do tabulky. Použitím zkrácené notace funkce **Part** docílíme toho, že se zobrazí ve výstupu **Manipulate** pouze tabulka se základními informacemi o intervalu, který odpovídá vybranému souboru.

```
TableForm[elements[[i]], TableAlignments → {Left},
 TableHeadings → {{"tMIN[°C]", "tMAX[°C]", "tMEAN[°C]",
  "časPOČÁTEČNÍ", "časKONCOVÝ"}, None}, TableSpacing → {2, 2}]
```

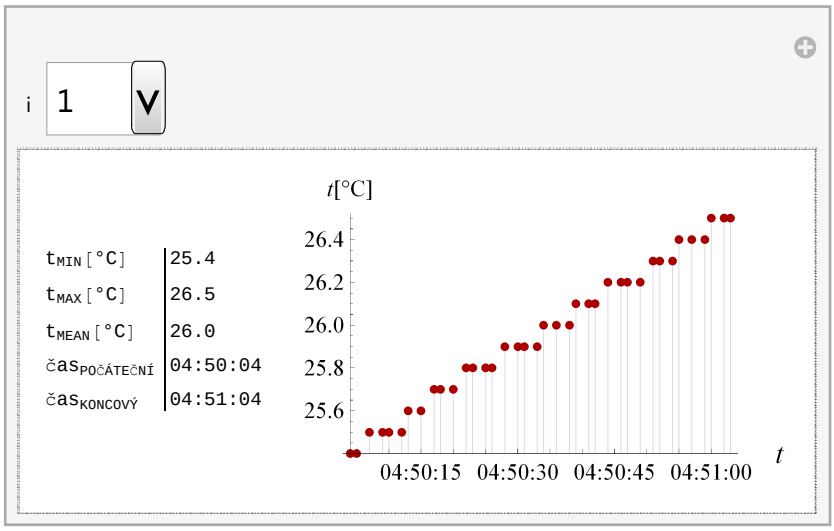
Dalším použitím **TableForm** zobrazíme vybrané dvojice prvků z *listu* **tab** a *listu* **grafy** v **Manupulate** jako tabulku 1x2.

```
TableForm[
 {TableForm[elements[[i]], TableAlignments → {Left}, TableHeadings →
   {{"tMIN[°C]", "tMAX[°C]", "tMEAN[°C]", "časPOČÁTEČNÍ", "časKONCOVÝ"},
    None}, TableSpacing → {2, 2}], grafy[[i]]},
 TableDirections → Row, TableSpacing → {8, 5}]
```

Tuto tabulku vložíme do funkce **Manipulate**. Možností (*option*) ControlType zvolíme typ kontroléru, který bude manipulovat s grafy a tabulkami. *Mathematica* nabízí různé druhy kontrolérů, například posuvník (*Slider*), animátor (*Animator*), manipulátor (*Manipulator*), rozbalovací nabídku (*PopupMenu*), přepínací lištu (*TogglerBar*), lištu zaškrtávacích polí (*CheckBoxBar*) a jiné. V našem případě jsme zvolili rozbalovací nabídku.

In[79]:=
```
Manipulate[
  TableForm[{TableForm[elements[[i]], TableAlignments → {Left},
     TableHeadings → {{"tMIN[°C]", "tMAX[°C]", "tMEAN[°C]",
        "časPOČÁTEČNÍ", "časKONCOVÝ"}, None}, TableSpacing → {2, 2}],
    grafy[[i]]}, TableDirections → Row, TableSpacing → 3],
  {i, Range[Length[grafy]], ControlType -> PopupMenu}] //
 Style[#, 10] &
```

Out[79]=



Obr. 6: Interaktivní výstup funkce ***Manipulate****.*

Výraz **{i,Range[Length[grafy]],ControlType->PopupMenu}** nám říká, že proměnná **i** bude nabývat hodnot v rozsahu čísel od 1 do délky *listu* **grafy**. Zároveň bude tento rozsah čísel zakomponován do rozbalovací nabídky. Výběrem libovolného čísla z této nabídky se zobrazí dvojice tabulka - graf. Zvolené číslo vyjadřuje pozici tabulky v *listu* **elements**, resp. grafu v *listu* **grafy**. Záměnou *listu* **{i,1,"Výběr souboru"}** za **i** se přejmenuje rozbalovací nabídka. Prvky v rozbalovací nabídce lze zaměnit tím, že každé pozici přidělíme výraz pomocí funkcí **ReplaceAll**, **Rule**, **Table**, **Part**, **Length**, **Drop**, a **Flatten**.

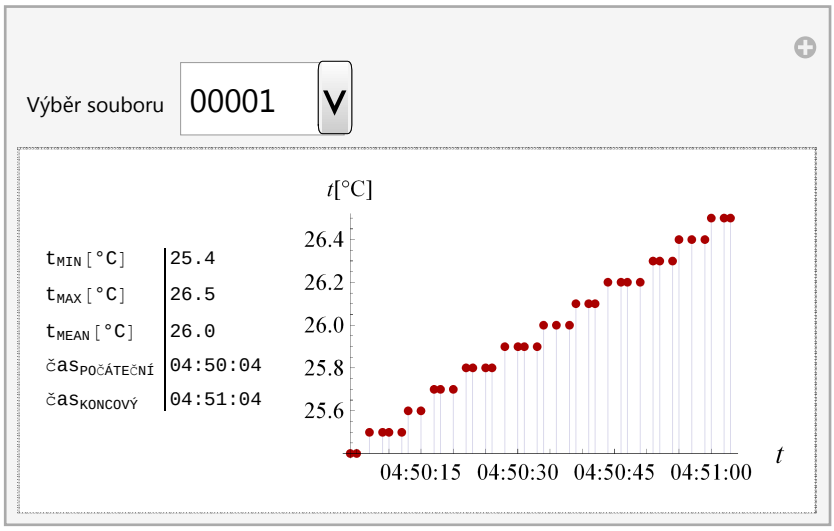
In[80]:=
```
prirazeni = Flatten[Table[{p, Drop[vysledek, 1][[p, 1]]},
   {p, 1, Length[Drop[vysledek, 1]]}] /. {a_, b_} → Rule[a, b]]
```

Out[80]=
```
{1 → 00001, 2 → 00002, 3 → 00003, 4 → 00004, 5 → 00005, 6 → 00006,
 7 → 00007, 8 → 00008, 9 → 00009, 10 → 00010, 11 → 00011, 12 → 00012,
 13 → 00013, 14 → 00014, 15 → 00015, 16 → 00016, 17 → 00017, 18 → 00018,
 19 → 00019, 20 → 00020, 21 → 00021, 22 → 00022, 23 → 00023, 24 → 00024,
 25 → 00025, 26 → 00026, 27 → 00027, 28 → 00028, 29 → 00029, 30 → 00030,
 31 → 00031, 32 → 00032, 33 → 00033, 34 → 00034, 35 → 00035}
```

List **prirazeni** dosadíme do funkce **Manipulate** místo *výrazu* **Range[Length[grafy]]**.

In[81]:=
```
Manipulate[
  TableForm[{TableForm[elements[[i]], TableAlignments → {Left},
     TableHeadings → {{"t_MIN [°C]", "t_MAX [°C]", "t_MEAN [°C]",
        "čas_POČÁTEČNÍ", "čas_KONCOVÝ"}, None}, TableSpacing → {2, 2}],
    grafy[[i]]}, TableDirections → Row, TableSpacing → 3],
  {{i, 1, "Výběr souboru"}, prirazeni,
   ControlType -> PopupMenu}] // Style[#, 10] &
```

Out[81]=



t_{MIN} [°C]	25.4
t_{MAX} [°C]	26.5
t_{MEAN} [°C]	26.0
$čas_{POČÁTEČNÍ}$	04:50:04
$čas_{KONCOVÝ}$	04:51:04

Obr. 7: Interaktivní výstup funkce ***Manipulate****.*

Souhrnnou tabulku (tab. 4) připojíme k interaktivní dvojici tabulka - graf. Opět použijeme funkci **TableForm**.

```
TableForm[
 {TableForm[{TableForm[elements[[i]], TableAlignments → {Left},
     TableHeadings → {{"t_MIN [°C]", "t_MAX [°C]", "t_MEAN [°C]",
        "čas_POČÁTEČNÍ", "čas_KONCOVÝ"}, None}, TableSpacing → {2, 2}],
    grafy[[i]]}, TableDirections → Row, TableSpacing → 3],
  tabulka}, TableSpacing → {5, 5}]
```

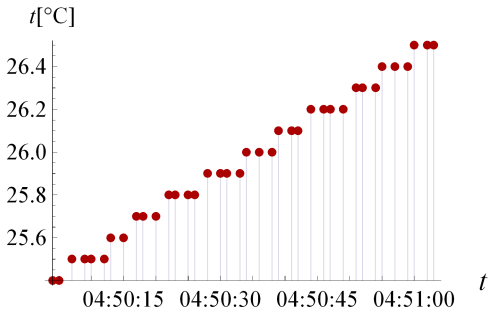
Tento předpis tabulky dosadíme do **Manipulate** a doplníme o možnost exportu tabulky kliknutím na tlačítko (*Button*).

In[82]:=
```
Manipulate[TableForm[
  {TableForm[{TableForm[elements[[i]], TableAlignments → {Left},
     TableHeadings → {{"t_MIN [°C]", "t_MAX [°C]", "t_MEAN [°C]",
        "čas_POČÁTEČNÍ", "čas_KONCOVÝ"}, None}, TableSpacing → {2, 2}],
    grafy[[i]]}, TableDirections → Row, TableSpacing → 5],
  tabulka, Button["Export souhrnné tabulky",
   Export["Hempsod05_320.dat", vysledek, Alignment → Right],
   ImageSize → 200]},
 TableAlignments → Center, TableSpacing → 2],
{{i, 1, "Výběr souboru"}, prirazeni,
 ControlType -> PopupMenu}] // Style[#, 10] &
```

Out[82]=

Výběr souboru 00001

t_{MIN} [°C]	25.4
t_{MAX} [°C]	26.5
t_{MEAN} [°C]	26.0
$čas_{POČÁTEČNÍ}$	04:50:04
$čas_{KONCOVÝ}$	04:51:04



č.s.	MIN[℃]	MAX[℃]	MEAN[℃]	SOUBOR	DATUM	ZAČÁTEK	KONEC
00001	25.4	26.5	26.0	hempsod05_320	06–Oct–2013	04:50:04	04:51:04
00002	26.7	27.7	27.2	hempsod05_320	06–Oct–2013	04:51:15	04:52:15
00003	30.8	225.8	118.5	hempsod05_320	06–Oct–2013	04:52:25	04:53:25
00004	259.4	312.8	295.9	hempsod05_320	06–Oct–2013	04:53:35	04:54:35
00005	314.7	320.6	318.2	hempsod05_320	06–Oct–2013	04:54:45	04:55:45
00006	321.2	323.2	322.3	hempsod05_320	06–Oct–2013	04:55:55	04:56:55
00007	324.1	324.3	324.2	hempsod05_320	06–Oct–2013	04:58:34	04:59:34
00008	323.4	324.0	323.7	hempsod05_320	06–Oct–2013	04:59:45	05:00:45
00009	322.8	323.4	323.1	hempsod05_320	06–Oct–2013	05:00:55	05:01:55
00010	321.8	322.6	322.2	hempsod05_320	06–Oct–2013	05:02:05	05:03:05
00011	321.7	321.9	321.8	hempsod05_320	06–Oct–2013	05:03:15	05:04:15
00012	321.5	321.7	321.5	hempsod05_320	06–Oct–2013	05:04:26	05:05:26
00013	321.3	321.5	321.4	hempsod05_320	06–Oct–2013	05:05:36	05:06:36
00014	320.7	321.2	321.0	hempsod05_320	06–Oct–2013	05:06:46	05:07:46
00015	320.6	320.9	320.7	hempsod05_320	06–Oct–2013	05:07:57	05:08:57
00016	320.4	320.7	320.5	hempsod05_320	06–Oct–2013	05:09:08	05:10:08
00017	320.3	320.5	320.4	hempsod05_320	06–Oct–2013	05:10:18	05:11:18
00018	320.1	320.5	320.3	hempsod05_320	06–Oct–2013	05:11:28	05:12:28
00019	320.1	320.3	320.2	hempsod05_320	06–Oct–2013	05:12:39	05:13:39
00020	320.2	320.5	320.3	hempsod05_320	06–Oct–2013	05:13:49	05:14:49
00021	320.6	320.7	320.6	hempsod05_320	06–Oct–2013	05:14:59	05:15:59
00022	320.3	320.5	320.4	hempsod05_320	06–Oct–2013	05:16:09	05:17:09
00023	320.3	320.5	320.4	hempsod05_320	06–Oct–2013	05:17:20	05:18:20
00024	320.1	320.3	320.2	hempsod05_320	06–Oct–2013	05:18:31	05:19:31
00025	320.1	320.3	320.2	hempsod05_320	06–Oct–2013	05:19:41	05:20:41
00026	320.2	320.5	320.4	hempsod05_320	06–Oct–2013	05:20:52	05:21:52
00027	320.5	320.6	320.6	hempsod05_320	06–Oct–2013	05:22:02	05:23:02
00028	320.4	320.6	320.5	hempsod05_320	06–Oct–2013	05:23:12	05:24:12
00029	320.6	320.7	320.7	hempsod05_320	06–Oct–2013	05:24:22	05:25:22
00030	320.6	320.7	320.7	hempsod05_320	06–Oct–2013	05:25:33	05:26:33
00031	320.7	320.8	320.7	hempsod05_320	06–Oct–2013	05:26:43	05:27:43
00032	320.8	320.8	320.8	hempsod05_320	06–Oct–2013	05:27:53	05:28:53
00033	320.5	326.9	322.5	hempsod05_320	06–Oct–2013	05:29:04	05:30:04
00034	327.8	329.7	329.0	hempsod05_320	06–Oct–2013	05:30:14	05:31:14
00035	329.8	329.9	329.9	hempsod05_320	06–Oct–2013	05:31:24	05:32:24

Export souhrnné tabulky

Obr. 8: Interaktivní výstup funkce ***Manipulate****.*

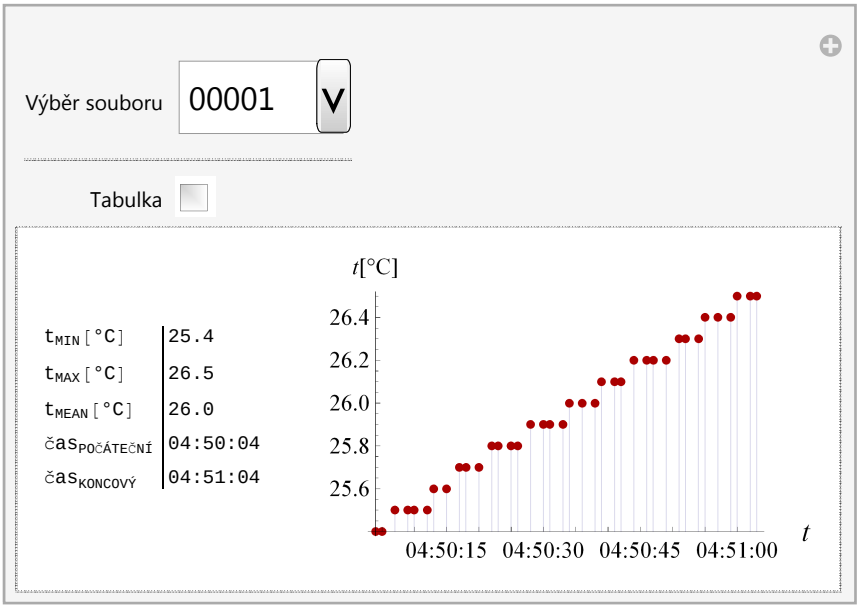
Kontrolér **CheckBox** (zaškrtávací políčko) pracuje na bázi pravda/nepravda (**True**/**False**). Toho využijeme pro zobrazení/skrytí souhrnné tabulky (tab. 4) ve výstupu funkce **Manipulate**. Obecná forma zápisu je následující

```
Manipulate[If[Název políčka, struktura1, struktura2], možnosti,
{Název políčka, {False, True}}]
```

Výraz **struktura1** vyjadřuje vstup funkce **Manipulate** v případě, že je políčko v režimu pravda (**True**). Naopak **struktura2** se vyhodnotí tehdy, když zaškrtávací políčko má hodnotu nepravda (**False**). Pořadí v *listu* **{False,True}** je podstatné. V tomto případě je primární výstup funkce **Manipulate** nastaven na hodnotu **False**. Znamená to, že se vyhodnotí **struktura2**. **Struktura1** se vyhodnotí až v případě zaškrtnutí políčka. Pokud bychom pořadí otočili, primárním výstupem **Manipulate** by byla **struktura1**.

In[83]:=
```
Manipulate[If[Tabulka, TableForm[
    {TableForm[{TableForm[elements[[i]], TableAlignments → {Left},
        TableHeadings → {{"tMIN[°C]", "tMAX[°C]", "tMEAN[°C]",
          "časPOČÁTEČNÍ", "časKONCOVÝ"}, None}, TableSpacing → {2, 2}],
      grafy[[i]]}, TableDirections → Row, TableSpacing → 5],
   tabulka, Button["Export souhrnné tabulky",
    Export["Hempsod05_320.dat", vysledek, Alignment → Right],
    ImageSize → 200]},
  TableAlignments → Center, TableSpacing → 2],
 TableForm[{TableForm[elements[[i]], TableAlignments → {Left},
    TableHeadings → {{"tMIN[°C]", "tMAX[°C]", "tMEAN[°C]",
      "časPOČÁTEČNÍ", "časKONCOVÝ"}, None}, TableSpacing → {2, 2}],
   grafy[[i]]}, TableDirections → Row, TableSpacing → 5]],
 {{i, 1, "Výběr souboru"}, prirazeni,
  ControlType -> PopupMenu},
 Delimiter, {Tabulka, {False, True}}] //
Style[#, 10] &
```

Out[83]=



*Obr. 9: Interaktivní výstup funkce **Manipulate**.*

4. 2 Obecný postup zpracování všech měření

Funkcemi **NotebookDirectory** a **SetDirectory** nastavíme cestu k tomuto *notebooku*.

```
SetDirectory[NotebookDirectory[]];
```

Import více souborů současně umožňuje kombinace funkcí **FileNames**, **Import**, **Map** a *Pure Function*.

In[84]:=
```
im = Map[Import[#, "Table"] &, FileNames["DESY\\*txt"]]
```

Out[84]=

A very large output was generated. Here is a sample of it:

```
{{{date/time, sec., from, start, Temp, 1, Temp, 2, SETpoint},
            {6.10.2013, 22:16:29, 0, 0., 41.9, 0},
            {6.10.2013, 22:16:31, 2, 0., 42.5, 0},
            {6.10.2013, 22:16:33, 3, 0., 42.6, 0},
            {6.10.2013, 22:16:34, 5, 0., 42.6, 0},
            {6.10.2013, 22:16:36, 7, 0., 42.6, 0},
          {6.10.2013, 22:16:37, 8, 0., 42.6, 0}, <<40>>,
            {6.10.2013, 22:17:44, 74, 0., 42.9, 0},
            {6.10.2013, 22:17:45, 76, 0., 43., 0},
            {6.10.2013, 22:17:47, 78, 0., 43., 0},
            {6.10.2013, 22:17:48, 79, 0., 42.9, 0},
            {6.10.2013, 22:17:50, 81, 0., 43., 0},
            {6.10.2013, 22:17:52, 82, 0., 43., 0}},
            <<53>>, {{<<1>>}, <<1582>>, {<<1>>}}}
```

Show Less | Show More | Show Full Output | Set Size Limit...

Z *listu* **im** odstraníme hlavičky jednotlivých souborů, změníme formát času a vybereme z nich pouze data, časy a teploty.

In[85]:=
```
Teploty =
 ReplaceRepeated[Map[ReplaceAll[#, {"4.10.2013" → "4-Oct-2013",
       "5.10.2013" → "5-Oct-2013", "6.10.2013" → "6-Oct-2013",
       "7.10.2013" → "7-Oct-2013", "8.10.2013" → "8-Oct-2013"}] &,
   Map[DeleteCases[#, {"date/time", "sec.", "from", "start",
       "Temp", 1, "Temp", 2, "SETpoint"}] &, im]],
  {a_, b_, c_, d_, e_, f_} → {a, b, e}]
```

Out[85]=

A very large output was generated. Here is a sample of it:

```
{{{6-Oct-2013, 22:16:29, 41.9}, {6-Oct-2013, 22:16:31, 42.5},
  {6-Oct-2013, 22:16:33, 42.6}, {6-Oct-2013, 22:16:34, 42.6},
  {6-Oct-2013, 22:16:36, 42.6}, {6-Oct-2013, 22:16:37, 42.6},
  {6-Oct-2013, 22:16:39, 42.6}, {6-Oct-2013, 22:16:41, 42.6},
            {6-Oct-2013, 22:16:42, 42.6}, <<35>>,
  {6-Oct-2013, 22:17:40, 43.}, {6-Oct-2013, 22:17:42, 43.},
  {6-Oct-2013, 22:17:44, 42.9}, {6-Oct-2013, 22:17:45, 43.},
  {6-Oct-2013, 22:17:47, 43.}, {6-Oct-2013, 22:17:48, 42.9},
  {6-Oct-2013, 22:17:50, 43.}, {6-Oct-2013, 22:17:52, 43.}},
            <<53>>, {{<<1>>}, <<1581>>, {<<1>>}}}
```

Show Less | Show More | Show Full Output | Set Size Limit...

Funkce **FileBasedName** separuje z cesty k *notebooku* pouze název souboru.

In[86]:=
```
MeasureName = Map[FileBaseName, FileNames["DESY\\*.txt"]];
```

```
Take[%, 5]
```

```
{4Na457Fe4_1NaFeO2m050_345,
 CNa457Fe4_JNa57FeO2m050_345, CNa457Fe4_JNaFeO2m050_345,
  CNa4Fe4_JNa57FeO2m050_345, CNaFe_JNa57FeO2_from50}
```

Uvádíme ukázku prvních pěti souborů. Znak **%** v argumentu funkce **Take** má význam hodnoty posledního vyhodnoceného výrazu.

Souhrnné zpracování všech měření je speciální případ, pro který nejsou v programu *Mathematica* přímo definované funkce. Jako vzor k jejich vytvoření použijeme konstrukci složených funkcí, jejichž popis jsme uvedli v oddílu 3. 3. 2.

První funkce je definovaná pro převedení dat a časů na absolutní čas.

In[87]:=
```
DataFCE[teploty_List] :=
 (time = Flatten[Partition[ReplaceRepeated[teploty,
      {o_, p_, q_} → {o, p}], Length[teploty]], 1];
  temperature = ReplaceRepeated[teploty, {o_, p_, q_} → {q}];
  absoluteTime = Map[AbsoluteTime[#] &, Map[ToString[#] &, time]];
  Thread[{absoluteTime, Flatten[temperature]}])
```

Druhá funkce zpracuje *list* načtených souborů *.fio. Její výstup má formu *listu* dvojic počátečních a koncových časů.

In[88]:=
```
TimeFCE[Time_List] := (finiteTime = Map[AbsoluteTime[#] &,
    Table[Time[[i]][[6, 3]], {i, 1, Length[Time]}]];
  initialTime = Map[# - 60 &, finiteTime];
  G = Thread[{initialTime, finiteTime}];
  DeleteCases[G, {-60 + $Failed, $Failed}])
```

Funkce **fce** rozdělí po dosazení *list* **Teploty** na intervaly.

In[89]:=
```
fce[teploty_List, casy_List] :=
 (A = (B = Table[Flatten[Position[Table[((casy[[j, 1]] ≤
          teploty[[i, 1]]) && (teploty[[i, 1]] ≤ casy[[j, 2]])),
       {i, 1, Length[teploty]}], True]], {j, 1, Length[casy]}];
   DeleteCases[B, {}]); Table[Part[teploty, A[[j, i]]],
  {j, 1, Length[A]}, {i, 1, Length[A[[j]]]}])
```

Další funkce zpracuje výstupy funkce **fce** do grafů

In[90]:=
```
grafyFCE[intervaly_List] :=
 Map[DateListPlot[#, Axes → True, Frame → False,
    GridLines → None, Filling → Axis, ImageSize → 250,
    PlotStyle → Directive[PointSize[Medium], Darker[Red]],
    AxesLabel → {Style["   t", 18], Style["t[°C]", 15]},
    LabelStyle → Directive[Black, 12]] &, intervaly]
```

Výstup funkce **tabulkaTableFCE** je *list* čísel souborů, minimálních, maximálních, průměrných teplot, názvů souborů, dat, počátečních, koncových časů. Obsahuje i popisky jednotlivých sloupců. Na místa argumentů se dosadí listy časů, intervalů a umístění souborů *.fio.

```
In[91]:= tabulkaTableFCE[cas_List, intervaly_List, soubory_List] :=
 (P = Table[Flatten[ReplaceRepeated[intervaly[[i]], {a_, b_} :> {b}]],
    {i, 1, Length[intervaly]}];
  max = Map[NumberForm[#, {4, 1}] &, Map[Max, P]];
  min = Map[NumberForm[#, {4, 1}] &, Map[Min, P]];
  mean = Map[NumberForm[#, {4, 1}] &, Map[Mean, P]];
  poc = Table[DateString[cas[[i, 1]],
     {"Hour", ":", "Minute", ":", "Second"}], {i, 1, Length[cas]}];
  konc = Table[DateString[cas[[i, 2]],
     {"Hour", ":", "Minute", ":", "Second"}], {i, 1, Length[cas]}];
  datum = Table[DateString[cas[[i, 1]], {"Day", "-",
      "MonthNameShort", "-", "Year"}], {i, 1, Length[cas]}];
  nazvysouboru = StringDrop[Flatten[soubory], -6];
  cislasouboru = StringTake[Flatten[soubory], -5];
  Q = Table[Join[{cislasouboru[[i]], min[[i]], max[[i]], mean[[i]],
      nazvysouboru[[i]], datum[[i]], poc[[i]], konc[[i]]}],
    {i, 1, Length[min]}]; Prepend[Q, {"č.s.", "MIN[°C]",
    "MAX[°C]", "MEAN[°C]", "SOUBOR", "DATUM", "ZAČÁTEK", "KONEC"}])
```

Funkce **tabulkaGridFCE** vytvoří souhrnnou tabulku.

```
In[92]:= tabulkaGridFCE[vysledek_List] :=
  Grid[vysledek, Frame → All] // Style[#, FontFamily → "Arial", 10] &
```

Poslední funkcí je **manipulateFCE**. Její výstup je interaktivní a obsahuje graf, tabulku se základními informacemi o grafu, přehlednou tabulku intervalů a tlačítko pro export této tabulky do souboru.

```
In[93]:= manipulateFCE[vysledek_List, grafy_List, nazevtabulky_String] :=
 (prirazeni = Flatten[Table[{i, Drop[vysledek, 1][[i, 1]]},
       {i, 1, Length[Drop[vysledek, 1]]}] /. {a_, b_} -> Rule[a, b]];
  Manipulate[If[Tabulka, TableForm[
     {TableForm[{TableForm[ReplaceRepeated[Drop[vysledek, 1],
           {a_, b_, c_, d_, e_, f_, g_, h_} -> {b, c, d, g, h}]][[i]],
        TableAlignments → {Left}, TableHeadings → {{"t_MIN[°C]",
           "t_MAX[°C]", "t_MEAN[°C]", "čas_POČÁTEČNÍ", "čas_KONCOVÝ"}, None},
        TableSpacing → {2, 2}], grafy[[i]]},
      TableDirections → Row, TableSpacing → 3],
     Style[tabulkaGridFCE[vysledek], 10, FontFamily → "Arial"],
     Button["Export souhrnné tabulky", Export[nazevtabulky,
       vysledek, Alignment → Right], ImageSize → 200]},
    TableSpacing → 2, TableAlignments → {Center}], TableForm[
    {TableForm[ReplaceRepeated[Drop[vysledek, 1],
        {a_, b_, c_, d_, e_, f_, g_, h_} -> {b, c, d, g, h}][[i]],
      TableAlignments → {Left}, TableHeadings → {{"t_MIN[°C]",
         "t_MAX[°C]", "t_MEAN[°C]", "čas_POČÁTEČNÍ", "čas_KONCOVÝ"}, None},
      TableSpacing → {2, 2}], grafy[[i]]},
    TableDirections → Row, TableSpacing → 3]],
   {{i, 1, "Výběr souboru"}, prirazeni,
    ControlType → PopupMenu},
   Delimiter, {Tabulka, {False, True}}]) //
  Style[#, 10, FontFamily → "Arial"] &
```

Dále budeme každý soubor zpracovávat jednotlivě. Důvodem jsou odlišné názvy textových souborů a odpovídajících sad *.fio souborů, které jsou rozmístěny

do různých podsložek. Tyto sady budeme tedy vyhledávat manuálně. Postup zpracování provedeme pro jedno měření (39.Na4Fe4m050_4325). Aplikace tohoto postupu na zbývající měření je záležitostí rutiny.

4. 2. 1 Zpracování souboru Na4Fe4m050_4325

Název 39. textového souboru, se kterým budeme pracovat, zjistíme z *listu* **MeasureName** pomocí zkrácené notace funkce **Part**.

```
In[94]:= MeasureName[[39]]
Out[94]= Na4Fe4m050_4325
```

- **Zpracování teplot**

Funkcí **Part** separujeme z *listu* **Teploty** 39. prvek. Výstupem je *list* absolutních časů a teplot. V tabulce 5 uvádíme prvních 5 hodnot, které jsme vybrali z *listu* **Teploty** funkcí **Take** a zobrazili do tabulky funkcí **TableForm**.

```
In[95]:= Data39 = DataFCE[Teploty[[39]]];
         Data39 //
           TableForm[#, TableHeadings → {Automatic, {"AbsoluteTime [s]",
                 "Temperature [°C]"}}, TableAlignments → Center] &;
         Take[Data39, 5] // TableForm[#, TableHeadings →
             {Automatic, {"AbsoluteTime [s]", "Temperature [°C]"}},
            TableAlignments → Center] &
```

Out[97]//TableForm=

	AbsoluteTime [s]	Temperature [°C]
1	3 589 957 495	59.
2	3 589 957 497	58.6
3	3 589 957 499	58.6
4	3 589 957 500	58.5
5	3 589 957 502	58.5

Tab. 5: Tabulka absolutních časů a teplot soouboru Na4Fe4m050_4325.txt.

- **Časy**

K tomu, abychom mohli vyhledat odpovídající sadu *.fio souborů, musíme znát krajní hodnoty časů v *listu* **Teploty** (funkce **Part** a **Length**).

```
In[98]:= Teploty[[39, 1]]
         Teploty[[39, Length[Teploty[[39]]]]]
Out[98]= {5-Oct-2013, 10:24:55, 59.}
Out[99]= {5-Oct-2013, 10:45:50, 433.7}
```

Umístění složky se správnými *.fio soubory vložíme do funkce **FileNames** a pomocí **Import**, **Map** a *Pure Function* importujeme do *notebooku*. Funkce **TimeFCE** má jako argument *list* importovaných souborů *.fio, jejím výstupem je *list* dvojic počátečních a koncových časů měření.

In[100]:=
```
Import39 = Map[Import[#, "Table"] &,
    FileNames["2013oct02\\sort\\Na4\\Na4Fe4m05_4325\\*.fio"]];
TimeData39 = TimeFCE[Import39];
Take[TimeData39, 5]
```

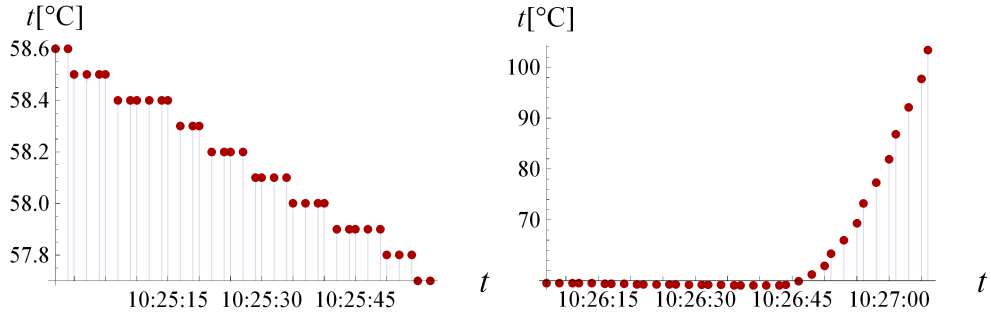
Out[102]=
```
{{3 589 957 497, 3 589 957 557},
 {3 589 957 567, 3 589 957 627}, {3 589 957 637, 3 589 957 697},
 {3 589 957 708, 3 589 957 768}, {3 589 957 778, 3 589 957 838}}
```

- **Intervaly a grafy**

Listy **Data39** a **TimeData39** dosadíme do funkce **fce**. Výstupem jsou intervaly časů a teplot, které odpovídají jednotlivým *.fio souborům. Tyto intervaly vyneseme pomocí funkce **grafyFCE** do grafů.

In[103]:=
```
Intervaly39 = fce[Data39, TimeData39];
Grafy39 = grafyFCE[Intervaly39];
Take[Grafy39, 2] // TableForm[#, TableDirections → Row] &
```

Out[105]//TableForm=



Obr. 10: Ukázka grafů časové závislosti teplot prvních dvou intervalů souboru Na4Fe4m050_4325.txt.

- **Souhrnná tabulka**

Funkce **tabulkaTableFCE** vytvoří *list* řádků souhrnné tabulky. Názvy souborů *.fio vypíšeme funkcemi **FileNames**, **FileBasedName**, **Map** a *Pure Function*.

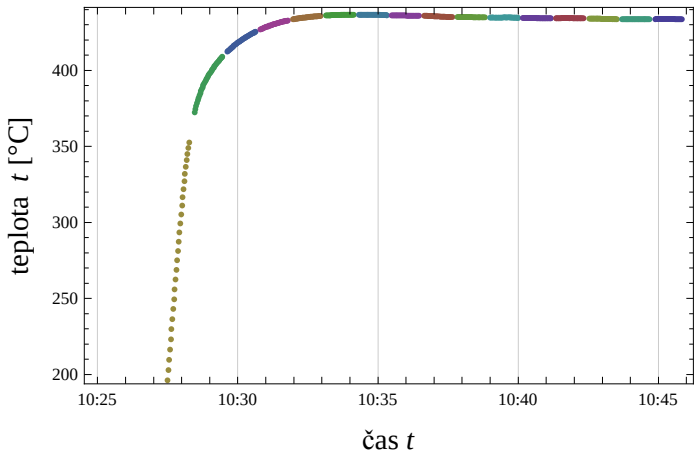
In[106]:=
```
Soubory39 = Map[FileBaseName,
    FileNames["2013oct02\\sort\\Na4\\Na4Fe4m05_4325\\*.fio"]];
Vysledek39 = tabulkaTableFCE[TimeData39, Intervaly39, Soubory39];
```

- **Manipulate**

Znázornění všech intervalů provedeme funkcí **DateListPlot**.

In[108]:=
```
DateListPlot[Intervaly39,
 FrameLabel → {Style["čas t", 15], Style["teplota  t [°C]", 15]}]
```

Out[108]=

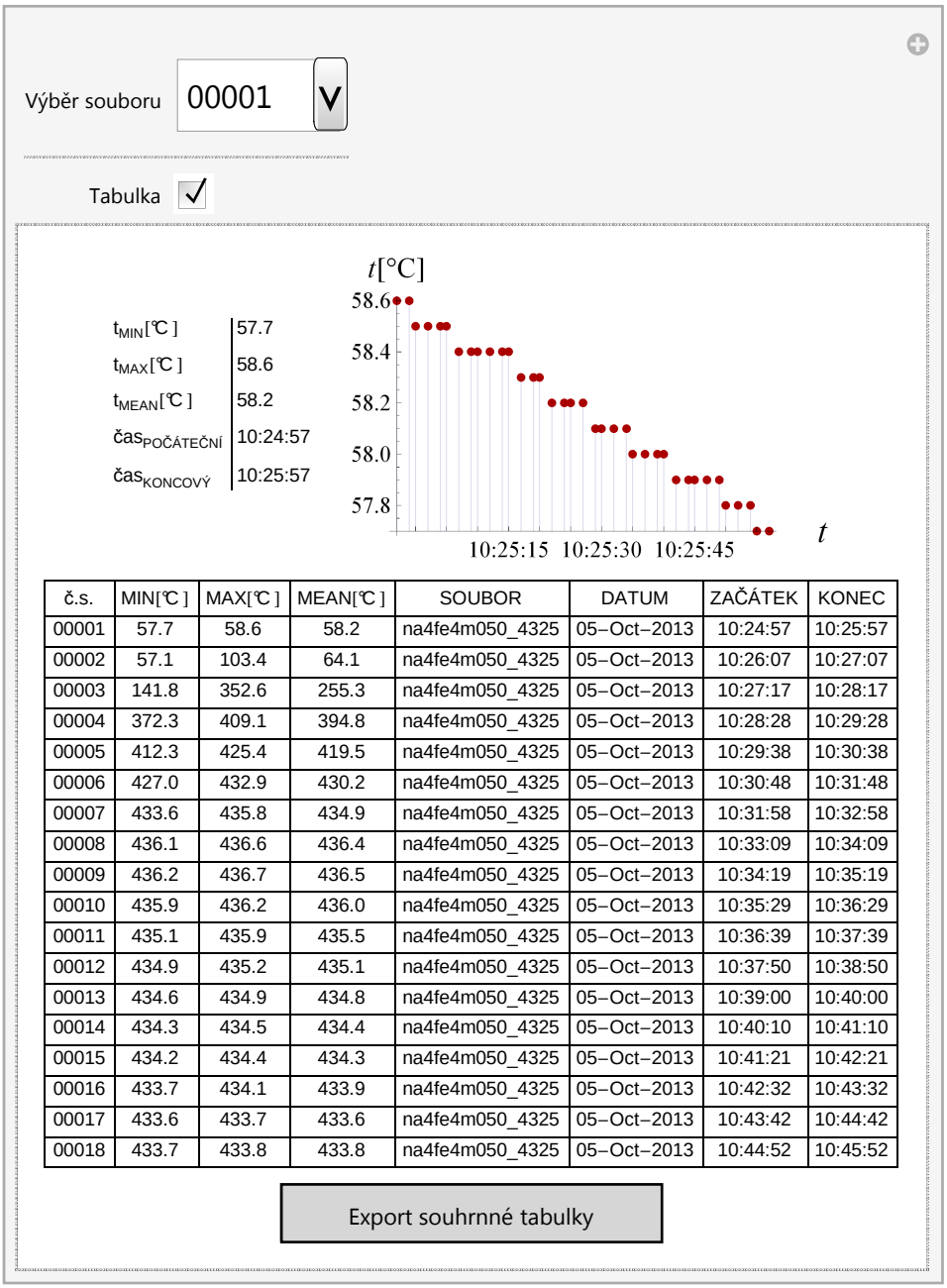


Obr. 11: Souhrnný graf závislosti.

Konečnou vizualizaci výsledků získáme jako výstup funkce manipulateFCE. Jejími argumenty jsou listy **`Vysledek39`**, **`Grafy39`** a název souboru (i s příponou) ve formě textového řetězce.

In[109]:=
```
manipulateFCE[Vysledek39, Grafy39, "Na4Fe4m050_4325.dat"]
```

Out[109]=



č.s.	MIN[℃]	MAX[℃]	MEAN[℃]	SOUBOR	DATUM	ZAČÁTEK	KONEC
00001	57.7	58.6	58.2	na4fe4m050_4325	05–Oct–2013	10:24:57	10:25:57
00002	57.1	103.4	64.1	na4fe4m050_4325	05–Oct–2013	10:26:07	10:27:07
00003	141.8	352.6	255.3	na4fe4m050_4325	05–Oct–2013	10:27:17	10:28:17
00004	372.3	409.1	394.8	na4fe4m050_4325	05–Oct–2013	10:28:28	10:29:28
00005	412.3	425.4	419.5	na4fe4m050_4325	05–Oct–2013	10:29:38	10:30:38
00006	427.0	432.9	430.2	na4fe4m050_4325	05–Oct–2013	10:30:48	10:31:48
00007	433.6	435.8	434.9	na4fe4m050_4325	05–Oct–2013	10:31:58	10:32:58
00008	436.1	436.6	436.4	na4fe4m050_4325	05–Oct–2013	10:33:09	10:34:09
00009	436.2	436.7	436.5	na4fe4m050_4325	05–Oct–2013	10:34:19	10:35:19
00010	435.9	436.2	436.0	na4fe4m050_4325	05–Oct–2013	10:35:29	10:36:29
00011	435.1	435.9	435.5	na4fe4m050_4325	05–Oct–2013	10:36:39	10:37:39
00012	434.9	435.2	435.1	na4fe4m050_4325	05–Oct–2013	10:37:50	10:38:50
00013	434.6	434.9	434.8	na4fe4m050_4325	05–Oct–2013	10:39:00	10:40:00
00014	434.3	434.5	434.4	na4fe4m050_4325	05–Oct–2013	10:40:10	10:41:10
00015	434.2	434.4	434.3	na4fe4m050_4325	05–Oct–2013	10:41:21	10:42:21
00016	433.7	434.1	433.9	na4fe4m050_4325	05–Oct–2013	10:42:32	10:43:32
00017	433.6	433.7	433.6	na4fe4m050_4325	05–Oct–2013	10:43:42	10:44:42
00018	433.7	433.8	433.8	na4fe4m050_4325	05–Oct–2013	10:44:52	10:45:52

Export souhrnné tabulky

Obr. 12: Interaktivní výstup funkce ***manipulateFCE****.*

5. Závěr

Cílem této bakalářské práce bylo zpracovat rozsáhlý soubor experimentálních dat v programu *Mathematica*, který nám poskytl možnost volby způsobu postupu. Zvolili jsme řešení pomocí funkcionálního programování v kombinaci s programováním rule-based, které je vhodné pro práci s *výrazy* typu *list*.

Průběh zpracování dat se odehrával v několika krocích. Nejdříve jsme do programu *Mathematica* importovali data obsažená v textových souborech (**In[47]**, **In[84]**) a upravili jejich strukturu. V této fázi se vyskytl problém s formátem časových údajů a způsobem jejich manipulace. Vše se vyřešilo jejich převodem na absolutní čas, se kterým se pracuje o poznání lépe (viz **In[53]** a **In[85]**).

V dalším kroku jsme importovali soubory *.fio, ze kterých jsme separovali pouze datum a čas zápisu daného měření (**In[55, 56]** a **In[100]**) do souboru *.fio. V případě ukázky zpracování dat souboru Hempsod05_320 problém nebyl, ten se vyskyl až tehdy, když bylo potřeba importovat soubory hromadně. Ty byly rozmístěny v různých podsložkách, které měly odlišný název od příslušných textových souborů. Komplikaci jsme vyřešili manuálním výběrem a importem každé složky souborů jednotlivě.

Od časů zápisu měření ve formátu absolutního času (**In[57]**) jsme odečetli 60 s a získali jsme čas začátku měření (**In[58]**). Pomocí těchto dvou údajů jsme *listy* s teplotami rozdělili na intervaly (**In[59]**, **In[103]**) a ty jsme pak vynesli do grafů (**In[66]**, obr. 5 a obr. 10). Pro každé měření jsme uvedli základní informace o jednotlivých intervalech do tabulky (**In[77]**, tab. 4 a **In[106]**).

Na závěr jsme grafy a tabulky použili jako argumenty funkce **Manipulate** (**In[83]**, obr.9 a **In[109]**, obr.12). Výstupem této funkce je interaktivní pole, kde je možné volbou čísla souboru daného měření přepínat mezi grafy a také zaškrtnutím políčka **Tabulka** zobrazit souhrnnou tabulku s možností jejího exportu do textového souboru.

Výstupy této práce se budou dále zpracovávat a použijí se při určování fázových složení a hyperjemných parametrů, které nesou informace o valenčním stavu, struktuře a magnetickém uspořádání studovaných materiálů. Úkolem celkové analýzy jednotlivých experimentů je zjistit, zda při reakci vznikají meziprodukty a určit rychlost dílčích procesů při rozkladu nebo přípravě vysokovalenčních stavů železa. Výsledky se uplatní ve výzkumu čištění znečištěných vod nebo také ve vývoji ekologicky méně náročných galvanických šlánků.

Literatura

- [1] PROCHÁZKA, V.: *Jaderný rezonanční rozptyl, Mössbauerova spektroskopie pomocí synchrotronového záření*, Univerzita Palackého v Olomouci, 2012.
- [2] GONSER, U.: *Mössbauer spektroscopy,* Springer-Verlag Berlin Heidelberg New York,. 1975.
- [3] SEDLÁK, B., KUZ'MIN, R. N.: *Jaderné resonanční metody ve fyzice pevných látek,* Státní pedagogické nakladatelství, Praha, 1978.
- [4] MULHAUPT, G., RÜFFER, R.: *Properties of Synchrotron radiation. Hyperfine Interactions*, 123/124: 13-30, 1999.
- [5] Deutsches Elektronen-Synchrotron: *Research facilities* [online], 2010, [Cit. 2014-04-25]. Dostupné z http://www.desy.de/research/facilities__projects/index_eng.html.
- [6] Anka - the Synchrotron Radiation Fasility at KIT [online], 2014-04-20, [Cit. 2014-04-25]. Dostupné z http://www.anka.kit.edu/28.php.
- [7] European Synchrotron Radiation Facility [online], 2013-06-18, [Cit. 2014-04-25]. Dostupné z http://www.esrf.eu/about.
- [8] Spring8 [online], 2014-04-15, [Cit. 2014-04-25]. Dostupné z http://www.spring8.or.jp/en.
- [9] Overview of the APS [online], 2014, [Cit. 2014-04-25]. Dostupné z http://www.aps.anl.gov/About/APS_Overview/.
- [10] Stephen Wolfram blog: *There Was a Time Before Mathematica* [online], 2013-06-06, [Cit. 2014-04-28]. Dostupné z http://blog.stephenwolfram.com/2013/06/there-was-a-time-before-mathematica/.
- [11] WolframAlpha [online], 2014, [Cit. 2014-04-29]. Dostupné z https://www.wolframalpha.com/.
- [12] Wolfram: What's New in *Mathematica* 9 [online], 2014, [Cit. 2014-04-29]. Dostupné z http://www.wolfram.com/mathematica/new-in-9/.
- [13] ABBASI, N. M., *A little bit of Mathematica history* [online], 2013, [Cit. 2014-04-29]. Dostupné z http : // 12000. org/my_notes/compare_mathematica/index.htm
- [14] DICK, S., RIDDLE, A., STEIN, D.: *Mathematica in the Laboratory*, Cambridge University Press, Cambridge, 1997.

- [15] WELLIN, P. R.,GAYLORD, R. J., KAMIN, S. N.: *An Introduction to Programming in Mathematica*, Third Edition, Cambridge University Press, NewYork, 2005.
- [16] Wolfram: *Wolfram Mathematica 9 Documentation Center* [online], 2014, [Cit. 2014-04-14]. Dostupné z http://reference.wolfram.com/mathematica/guide/Mathematica.html.

Příloha A. Hempsod05_320

```
In[110]:= SetDirectory[NotebookDirectory[]];

In[111]:= importteploty = Import["DESY\\Hempsod05_320.txt", "Table"];

In[112]:= deleteimport = DeleteCases[importteploty, {"date/time", "sec.",
            "from", "start", "Temp", 1, "Temp", 2, "SETpoint"}];

In[113]:= teploty = deleteimport //. {a_, b_, c_, d_, e_, f_} → {b, e};

In[114]:= DateListPlot[teploty, ImageSize → 350, AxesOrigin → {Automatic, 0},
            Filling → Bottom, LabelStyle → Directive[Bold, Medium],
            PlotLabel → "Hempsod_320", FrameLabel → {"Čas", "Teplota [°C]"}];

In[115]:= timeformat = deleteimport //. {a_, b_, c_, d_, e_, f_} → {a , b} /.
            "6.10.2013" → "6-Oct-2013";

In[116]:= absolutetime = Map[AbsoluteTime[#] &, Map[ToString[#] &, timeformat]];

In[117]:= data = Thread[
            {absolutetime, deleteimport //. {a_, b_, c_, d_, e_, f_} → e}];
```

```
In[118]:= timeimport = Map[Import[#, "Table"] &,
            FileNames["2013oct02\\sort\\HemPSod\\HemPSod05_320\\*.fio"]];

In[119]:= time = Map[Part[#, 6, 3] &, timeimport];

In[120]:= finitetime = Map[AbsoluteTime[#] &, time];

In[121]:= initialtime = Map[# - 60 &, finitetime];

In[122]:= timedata = Thread[{initialtime, finitetime}];
```

```
In[123]:= pozice =
            Table[Flatten[Position[Table[((timedata[[j, 1]] ≤ data[[i, 1]]) &&
                    (data[[i, 1]] ≤ timedata[[j, 2]])), {i, 1, Length[data]}],
                True]], {j, 1, Length[timedata]}] // DeleteCases[#, {}] &;

In[124]:= intervaly = Table[Part[data, pozice[[k, l]]],
            {k, 1, Length[pozice]}, {l, 1, Length[pozice[[k]]]}];

In[125]:= Length[intervaly]
Out[125]= 35

In[126]:= grafy = Map[DateListPlot[#, Axes → True, Frame → False,
              GridLines → None, Filling → Axis, ImageSize → 250,
              PlotStyle → Directive[PointSize[Medium], Darker[Red]],
              AxesLabel → {Style["    t", 15], Style["t[°C]", 12]},
              LabelStyle → Directive[Black, 12]] &, intervaly];
```

In[127]:=
```
temp = Table[ReplaceRepeated[intervaly[[m]], {a_, b_} → b],
   {m, 1, Length[intervaly]}];
min = Map[NumberForm[#, {4, 1}] &,
   Table[Min[temp[[n]]], {n, 1, Length[intervaly]}]];
max = Map[NumberForm[#, {4, 1}] &,
   Table[Max[temp[[n]]], {n, 1, Length[intervaly]}]];
mean = Map[NumberForm[#, {4, 1}] &,
   Table[Mean[temp[[n]]], {n, 1, Length[intervaly]}]];
initial = Map[DateString[#, {"Hour", ":", "Minute", ":", "Second"}] &,
   Table[timedata[[o, 1]], {o, 1, Length[timedata]}]];
finite = Map[DateString[#, {"Hour", ":", "Minute", ":", "Second"}] &,
   Table[timedata[[o, 2]], {o, 1, Length[timedata]}]];
datum = Map[DateString[#, {"Day", "-", "MonthNameShort", "-",
      "Year"}] &, Table[timedata[[o, 1]], {o, 1, Length[timedata]}]];
nazevsouboru = StringDrop[Flatten[Map[FileBaseName, FileNames[
      "2013oct02\\sort\\HemPSod\\HemPSod05_320\\*.fio"]]], -6];
cislasouboru = StringTake[Flatten[Map[FileBaseName, FileNames[
      "2013oct02\\sort\\HemPSod\\HemPSod05_320\\*.fio"]]], -5];
vysledek = Prepend[Table[Join[{cislasouboru[[i]], min[[i]],
      max[[i]], mean[[i]], nazevsouboru[[i]], datum[[i]],
      initial[[i]], finite[[i]]}], {i, 1, Length[max]}],
   {"č.s.", "MIN[°C]", "MAX[°C]", "MEAN[°C]", "SOUBOR",
    "DATUM", "ZAČÁTEK", "KONEC"}];
tabulka = Grid[vysledek, Frame → All] //
   Style[#, FontFamily → "Arial", 10] &;
```

In[138]:=
```
elements = ReplaceRepeated[Drop[vysledek, 1],
   {a_, b_, c_, d_, e_, f_, g_, h_} → {b, c, d, g, h}];
```

In[139]:=
```
prirazeni = Flatten[Table[{p, Drop[vysledek, 1][[p, 1]]},
     {p, 1, Length[Drop[vysledek, 1]]}] /. {a_, b_} → Rule[a, b]];
```

In[140]:=
```
Manipulate[If[Tabulka, TableForm[
     {TableForm[{TableForm[elements[[i]], TableAlignments → {Left},
         TableHeadings → {{"t_MIN[°C]", "t_MAX[°C]", "t_MEAN[°C]",
            "čas_POČÁTEČNÍ", "čas_KONCOVÝ"}, None}, TableSpacing → {2, 2}],
        grafy[[i]]}, TableDirections → Row, TableSpacing → 5],
      tabulka, Button["Export souhrnné tabulky",
       Export["Hempsod05_320.dat", vysledek, Alignment → Right],
       ImageSize → 150]},
     TableAlignments → Center, TableSpacing → 2], TableForm[
     {TableForm[{TableForm[elements[[i]], TableAlignments → {Left},
         TableHeadings → {{"t_MIN[°C]", "t_MAX[°C]", "t_MEAN[°C]",
            "čas_POČÁTEČNÍ", "čas_KONCOVÝ"}, None}, TableSpacing → {2, 2}],
        grafy[[i]]}, TableDirections → Row, TableSpacing → 5]}]],
   {{i, 1, "Výběr souboru"}, prirazeni,
    ControlType -> PopupMenu},
   Delimiter, {Tabulka, {False, True}}] //
  Style[#, 10] &;
```

Příloha B. Zpracování všech souborů

1. 4Na457Fe4_1NaFeO2m050_345
2. CNa457Fe4_JNa57FeO2m050_345
3. CNa457Fe4_JNaFeO2m050_345
4. CNa4Fe4_JNa57FeO2m050_345
5. CNaFe_JNa57FeO2_from50
6. Fe_57_200C
7. Fe_57_300C
8. Fe_57_400C
9. Fe_57_500C
10. Fe_57_600C
11. Fe5u_400W
12. Hempsd05_330_new_2
13. Hempsd05_330_new
14. Hempsod05_320
15. Hempsod05_3225
16. Hempsod05_325
17. Hempsod05_330
18. Hempsod05_340
19. Hempsod05_400
20. K3Fe5m050_750
21. K3Fe5m050_90

22. K3Fe5m050_s400
23. K3Fe5m050_s
24. K3FeO4_80
25. K3FeO4_m050_80new
26. Na457Fe4+Na57FeO2m050_410_2
27. Na457Fe4+Na57FeO2m050_410
28. Na457Fe4+NaFeO2m050_410
29. Na4Fe4_325
30. Na4Fe4_345
31. Na4Fe4_360
32. Na4Fe4M025_470
33. Na4Fe4m050_125
34. Na4Fe4m050_145
35. Na4Fe4m050_390
36. Na4Fe4m050_400
37. Na4Fe4m050_410
38. Na4Fe4m050_420
39. Na4Fe4m050_4325
40. Na4Fe4m050_435_2
41. Na4Fe4m050_435
42. Na4Fe4m050_440_2
43. Na4Fe4m050_440

44. Na4Fe4m050_450_2

45. Na4Fe4m050_450

46. Na4Fe4m050_from220

47. Na4Fe4m050_from50

48. Na4Fe4M05_490

49. Na4Fe4m075_420

50. Na4Fe4m075_425

51. Na4Fe4m150_430_2

52. Na4Fe4m150_430

53. Na4Fe4+Na57FeO2m050_410

54. TK357FeO4_JK57FeO2_80

55. TK357FeO4+JKFeO2

In[151]:=
```
MeasureName[[55]];
```

55.1 Zpracování teplot

In[152]:=
```
Data55 = DataFCE[Teploty[[55]]];
Data55 //
  TableForm[#, TableHeadings -> {Automatic, {"AbsoluteTime [s]",
       "Temperature [°C]"}}, TableAlignments -> Center] &;
```

55.2 Časy

In[154]:=
```
Teploty[[55, 1]];
Teploty[[55, Length[Teploty[[55]]]]];
```

In[156]:=
```
Import55 = Map[Import[#, "Table"] &, FileNames[
    "2013oct02\\sort\\K3_mix\\K357FeO3_KFeO2_80\\*.fio"]];
TimeData55 = TimeFCE[Import55];
```

55.3 Intervaly, grafy

In[158]:=
```
Intervaly55 = fce[Data55, TimeData55];
Grafy55 = grafyFCE[Intervaly55];
```

55.4 Souhrnná tabulka

In[160]:=
```
Soubory55 = Map[FileBaseName, FileNames[
    "2013oct02\\sort\\K3_mix\\K357FeO3_KFeO2_80\\*.fio"]];
Vysledek55 = tabulkaTableFCE[TimeData55, Intervaly55, Soubory55];
tabulkaGridFCE[Vysledek55];
```

55.5 Manipulate

In[163]:=
```
DateListPlot[Intervaly55, ImageSize -> 350];
```

In[164]:=
```
manipulateFCE[Vysledek55, Grafy55, "TK357FeO4+JKFeO2.dat"];
```