Úvod do matematického modelování

Jiří Hřebíček
Michal Škrdla

2006

Obsah

Kapitola 1

Úvod do matematického modelování a jeho členění

Matematické modelování [1, 2] proniklo do různých oborů přírodních, technických, ekonomických i sociálních věd a stalo se důležitým pomocníkem při modelování a simulacích systémů, analýzách a předvídání různých procesů, jevů, chování druhů a stavů společenstev.

Matematické modely poskytují srozumitelný popis všech relevantních faktorů dané situace a umožňují odhalit podstatné vztahy mezi prvky studovaného systému. Systémy v našem učebním textu uvažujeme jako abstrakce, které si lidé vytvářejí v procesu poznání (kognitivní limity). Za jistých podmínek lze za systém považovat (neúplnost výčtu) např.:

a) Reálný objekt (přirozený či umělý);

b) Projekt reálného objektu;

c) Proces, komplex procesů;

d) Problém, komplex problémů;

e) Soubor informačních, regulačních a řídících aktivit vztahujících se k a) – d) (informační systém, řídící systém, komunikační systém, regulační systém);

f) Abstraktní myšlenkovou konstrukci, výrokovou konstrukci, konstrukci matematických výrazů apod. zaváděném na a) – e);

g) Abstraktní myšlenkovou konstrukci, atd. vytvářenou bez přímého vztahu k a) – e).

Použití matematického modelu přináší řadu výhod:

- Umožňuje zjistit informace o chování systému, i když učinit závěry přímo z originálu je nemožné nebo obtížné.

- Urychluje proces poznání objektivní reality. Procesy, které ve skutečném systému probíhají pozvolna a dlouhodobě, lze pomocí modelu sledovat během jeho výpočtu, který závisí na použité informační a komunikační technologii (ICT). [3]

- Usnadňuje a racionalizuje proces poznání. Matematický model systému dává přehledná, stručná zobrazení objektivní reality a umožňuje postup při řešení problému podle potřeby uživatele. Modely vnášejí nové poznání do našeho myšlení. [4]

- Umožňuje variantní řešení, tj. propočet celé řady variant možných výsledků řešení.[5]

- Identifikuje vznik chybného poznání objektivní reality (na rozdíl od experimentu v reálném systému).

Matematické modelování získává v posledních letech velký význam zejména při výuce přírodovědných předmětů jak v univerzitním prostředí tak i na vysokých školách technického zaměření. Do matematického modelování, stejně jako i do jiných odvětví vědy proniklo již od šedesátých let minulého století využití výpočetní techniky, nyní se bez využití ICT neumíme matematické modelování představit. Požadavky na matematické výpočty – symbolické a numerické výpočty, na jejich vysokou přesnost, maximální vizualizaci a interaktivní komunikaci s řešitelem, atd. – vedly k vytvoření komplexních programových systémů jako jsou např. komerční systémy Derive od firmy Texas Instruments (`http://www.derive.com`) , Maple od Maplesoft Inc. (`http://www.maplesoft.com`), MathCAD od MathSoft Inc. (`http://www.mathsoft.com`), Mathematica od Wolfram Reasearch, Inc. (`http://www.wolfram.com`), MuPAD (`http://research.mupad.de/`) vyvíjený universitou Paderhorne a firmou SciFace Software GmbH (`http://www.sciface.com/`), atd., (podrobněji na `http://www.symbolicnet.org/www-sites.html`) nebo volně přístupné systémy (open source) jako je např. R (`http://www.symbolicnet.org/lit.html`). Tyto systémy lze využít během celého procesu identifikace, analýzy, vývoje, implementace, řešení a ověřování, případně modifikace matematického modelu. Volně přístupné zdroje literatury k této problematice lze nalézt na `http://www.symbolicnet.org/lit.html`.

Dalším významným požadavkem při modelování systémů reálného světa je neurčitost, která se objevuje během vývoje matematického modelu především měkkých systémů jak v matematické formulaci a jejich parametrech a fyzikálních, příp. matematických konstantách, tak i vstupních datech. Dále v mezerách ve znalostech o tom, jak je vytvářený model citlivý [6] na změnu parametrů i dat, případně tato neurčitost může být způsobena zjednodušením modelu objektivní reality do vhodných matematických výrazů, chybami v měření, nedostatkem zkoumaných dat nebo nedokonalostí použitých metod. To však současné ICT jsou schopny již řešit.

Ve většině případů se v současné době používají výše uvedené systémy na vysokých školách pro demonstraci probírané látky na cvičeních, případně jsou využity studenty při individuální přípravě a analýze problémů.

Charakteristickým rysem inovace výuky matematického modelování ve studijních programech universit v České republice, Evropské unie a v zemích OECD se v posledních deseti letech stává používání nových ICT v rámci budování nového vědního oboru „Computational Science“ a „Mathematical Modelling“ [3].

Cílem stále více vysokých škol je ovšem zařadit do výuky předmět seznamující studenty s prací v podobných systémech a jejich využití při návrhu, analýze a testování netriviálních matematických modelů. To je rovněž cílem tohoto učebního textu, kde nejprve popíšeme matematický model, metodologie matematického modelování a praktických příkladech ukážeme využití ICT Maple při jejich řešení.

1.1 Matematický model

Matematický model je abstraktní model, který využívá matematického zápisu k popisu chování systému. Matematický model transformuje model do matematického zápisu, který má následující výhody:

- formalizaci zápisu danou historickým vývojem,
- přesná pravidla pro manipulaci s matematickými symboly,
- možnost využití ICT pro zpracování vytvořeného modelu.

I přes velký potenciál matematického zápisu není možné popsat reálné systémy, objekty či procesy, které jsou velmi komplikované. Proto musíme nejdříve identifikovat nejdůležitější části zkoumaného systému, který budeme modelovat a ty musí vytvářený model popisovat. Ostatní prvky systému můžeme buď zcela vyloučit nebo podstatně zjednodušit.

Základní složky matematického modelu

Matematický model obvykle popisuje systém [7] s pomocí množiny proměnných a množiny rovnic, které určují vztahy mezi nimi. Hodnoty proměnných mohou být např. reálná nebo celá čísla, booleovské hodnoty nebo textové řetězce. Proměnné representují nějaké vlastnosti systému, např. u měřených výstupů systémů mohou být výstupní signály, vzorkovaná data, výskyt dané události či jevu (ano/ne), apod.

V každém matematickém modelu můžeme rozlišit tři základní skupiny objektů, ze kterých se model skládá. Jsou to:

1. proměnné a konstanty,
2. matematické struktury,
3. řešení.

1.1.1 Proměnné a konstanty

V matematickém modelu uvažujeme základní skupiny proměnných: rozhodovací (řídící) proměnné, vstupní (exogenní) proměnné, stavové proměnné, náhodné proměnné a výstupní (endogenní) proměnné.

Rozhodovací (řídící) proměnné Jsou obvykle známy jako nezávisle proměnné. Představují zpravidla nejdůležitější procesy modelovaného systému, které se v matematickém modelování nazývají aktivity nebo entity nebo rozhodovací proměnné. Příklad: V modelu $I = \frac{U}{R}$ představují U a R napětí a odpor v příslušných jednotkách. Těmito dvěma řídícími proměnnými je určen proud I;

Vstupní (exogenní) proměnné Ovlivňují daný systém a jejich hodnoty jsou determinovány mimo modelovaný systém;

Stavové proměnné Jsou závislé na ostatních proměnných (rozhodovacích, vstupních, náhodných a exogenních proměnných);

Náhodné proměnné Jsou obvykle určeny pravděpodobnostní funkcí (diskrétní proměnná) nebo hustotou pravděpodobnosti (spojitá proměnná) a představují neurčitost v modelu.

Výstupní (endogenní) proměnné Jejich hodnoty jsou určeny (generovány) systémem či jeho modelem.

Dále můžeme proměnné a konstanty v modelu uvažovat jako:

Proměnné a konstanty identifikované (pojmenované) Identifikovaná proměnná nebo konstanta představuje konkrétní vlastnost reálného objektu, pojmenovanou názvem a fyzikální jednotkou v níž se měří. Příklady: x_k je výměra pšenice ozimé v ha, x_r produkce pšenice ozimé na parcele „U křížku“ v čase t, náhodná doba čekání sedmé jednotky v systému hromadné obsluhy v pátém kanálu obsluhy v minutách, c_i, k vzdálenost dodavatele D_i od spotřebitele S_k v km.

Proměnné a konstanty neidentifikované (pomocné) Slouží pro formalizaci matematického zápisu, implementaci algoritmů apod. obvykle se uvažují v bezrozměrných jednotkách.

Nekontrolovatelné proměnné Představují procesy v systému, jejichž míry nelze zjistit (jedná se další typ neurčitosti). Příklady: Velikost míry inflace v chaotických a nestandardních podmínkách nelze popsat ani pomocí pravděpodobnosti ani pomocí fuzzy funkce. V modelech situací “ad hoc” jsou charakteristiky počasí nekontrolovatelné konstanty nebo proměnné, protože nelze využít počtu pravděpodobnosti pro jejich popis.

1.1.2 Matematické struktury (omezující podmínky)

V matematických modelech se matematické struktury nazývají omezující podmínky. Dělíme je podle použitého matematického aparátu z některého odvětví matematiky:

Analytické struktury Jedná se o objekty [8] z odvětví Matematické analýzy, Lineární algebry a dalších odvětví matematiky. Příklad: soustavy rovnic (lineární, nelineární, skalární, vektorové, diferenciální, integrální, maticové, atd.), soustavy nerovnic (lineární, nelineární, se smíšenými omezeními, atd.), funkce (elementární, složené, holomorfní, stochastické, fuzzy, atd.), funkcionály, atd.

Geometrické struktury Model je popsán grafickými prostředky: body, přímkami, rovinami, křivkami. Příklad: Geometrická interpretace a řešení úloh v modelech lineárního programování. Grafická interpretace rovnováhy nabídky a poptávky v ekonometrických modelech, atd.

Topologické struktury Modely jsou vytvářeny pomocí objektů matematické teorie grafů. Příklad: Modely maximálních toků v sítích, nejspolehlivější cesty v grafu/síti. Dopravní a distribuční systémy zobrazené grafem. Logistické systémy popsané pomocí grafů a schémat. Topologické modely lze zpravidla ekvivalentně zobrazovat pomocí tzv. incidenčních matic (tabulek, matic souslednosti, apod.).

Arteficiální struktury Modely jsou popsány prvky programovacího jazyka. Příklad: Model systému zásob popsaný vývojovým diagramem (simulačním jazykem SIMULA 67, objektově orientovaným jazykem Smalltalk, atd.).

Kvalitativní struktury Model je popsán pomocí kvalitativních rovnic, kvalitativních nerovností nebo vágně. Příklad: kvalitativní matice, kvalitativní graf, jazykový operátor „velmi“ v teorii fuzzy množin, atd.

Některé speciální a především již standardní struktury matematického modelu mají specifické názvy. Příklady: Cobb-Douglasova funkce. Účelová funkce. Podmínky nezápornosti. Lagreangova funkce. Wolfeho podmínky.

1.1.3 Řešení matematického modelu

Řešení matematického modelu dělíme podle hlediska cílů modelování:

Přípustné řešení, nepřípustné řešení řešení vyhovuje, řešení nevyhovuje omezujícím podmínkám.

Maximální řešení, minimální řešení řešení splňuje maximalizační nebo minimalizační cílovou podmínku.

Optimální řešení řešení vyhovuje nejlépe požadovanému cíli podle představ a požadavků řešitele (tj. nemusí být nutně maximální či minimální).

Výchozí řešení řešení zpravidla zadané odhadem nebo sestrojené vhodným jednoduchým algoritmem. Není optimální, používá se jako start v algoritmech typu „step by step“, které jsou založeny na postupném zlepšování výchozího řešení až do jeho optimálního tvaru.

Výsledné řešení řešení, které může být vybráno jako optimální. Výsledných řešení může být k disposici konečně nebo i nekonečně mnoho. Z množiny výsledných řešení (alternativ) vybírá řešitel řešení pro praxi nejvhodnější (optimální).

Alternativní řešení řešení, které je podle předem zadaných kriterií rovnocenné s jiným řešením. Příklad: Dvě strategie investic do vybavení podniku předpokládají sice různé technologie, ale garantují dosažení stejné výše zisku.

Aproximativní řešení řešení vyhovuje omezujícím podmínkám přibližně nebo se k přesnému řešení pouze přibližuje (zpravidla se požaduje, aby termín „přibližně“ byl vhodným způsobem determinován, např. byla známa velikost chyby, když řešení použijeme).

1.2 Klasifikace matematických modelů

Během identifikace a analýzy [9, 7] modelovaného systému je vhodné určit do jaké kategorie matematický model spadá, což nám umožní snadněji rozpoznat základní vlastnosti a strukturu hledaného modelu.

Podle toho zda zahrnujeme do modelu náhodné veličiny lze modely rozdělit do dvou skupin: *deterministických* a *stochastických modelů*. Dále lze tyto skupiny rozdělit dle vztahu k průběhu času (*dynamické, statické*) nebo spojitosti (*spojité, diskrétní*). Matematické modely jsou obvykle složeny z proměnných, které jsou abstrakcí hledaných složek systému a operátorů nad těmito proměnnými, které mohou representovat algebraické operace, funkce, funkcionály, diferenciální operátory, atd. Pokud operátory v matematickém modelu jsou lineární hovoříme o *lineárních modelech*, v opačném případě o *nelineárních modelech*.

Dále můžeme uvažovat modely se soustředěnými (u homogenních modelů) a distribuovanými parametry (u heterogenních modelů). Mezi těmito skupinami leží mnoho dalších typů modelů, dále tříděných podle mnoha dalších kriterií, které lze využít.

Matematické modely se používají prakticky ve všech vědách a rozvoj jednotlivých věd je na jejich využívání bezprostředně závislý. Stupeň matematizace vědního oboru je uznávaným měřítkem jeho kvality a zárukou rozvoje. V oblastech přírodních a fyzikálních věd, technice, ekonomii, managementu, marketingu, sociálních a společenských vědách se používá velké množství různých typů matematických modelů, které můžeme klasifikovat podle různých hledisek. Nejobecnější klasifikace dělí matematické modely do dvou skupin:

Modely deskriptivní Slouží k zobrazení prvků a vztahů v systému a k analýze základních vlastností systému. Nezajímá nás určité cílové chování systému, pouze systém sám o sobě. Pomocí těchto typů modelů se odvozují další vlastnosti systému, určuje se jeho rovnovážný stav, stabilní stav, vliv změn uvnitř i ve vnějším okolí systému na jeho chování. Příklady: Rovnice $E = mc^2$, soustava diferenciálních rovnic modelující procesy zrodu a úmrtí, simulační model modelující výskyt škůdců porostu, rovnice nabídky a poptávky v konkurenčním prostředí, ekonometrický meziodvětvový model „Input-Output“, atd.

Modely normativní Slouží k analýze a řízení systému tak, aby byl splněn nějaký cíl nebo množina cílů. Zajímá nás cílové chování systému. Normativní model bývá často doplněn tzv. cílovou (účelovou) funkcí nebo soustavou takových funkcí. Nutnou součástí normativního modelu je extremální (minimální/maximální) řešení, které dává návod, jak požadovaného cíle (resp. cílů) dosáhnout. Normativní modely, jejichž cílem je nalezení optimálního řešení, se nazývají optimalizační modely.

Modely deskriptivní i normativní jsou dále děleny podle typu systému, k jehož modelování slouží, nebo podle typu matematických složek (proměnné, struktury, řešení) jež obsahují.

Modely statické Model zobrazuje a analyzuje systém bez zřetele k jeho časovému vývoji. Zobrazení se týká zpravidla určitého časového intervalu (týden, měsíc, rok, apod.).

Modely dynamické Model zobrazuje a analyzuje systém v průběhu času. Zobrazení může být typu „ex post“ nebo „ex ante“ a respektovat krátký či delší časový horizont.

Modely dynamizované Zpravidla se jedná o vyjádření časového prvku ve statickém modelu pomocí speciálních modelových technik. Dynamizované modely se používají v případě, kdy odpovídající dynamický model je velmi složitý nebo jej nedovedeme soudobými modelovými technikami spolehlivě konstruovat.

Modely deterministické Všechny proměnné, konstanty a funkce v modelu jsou deterministické (nenáhodné) veličiny nebo funkce.

Modely stochastické Alespoň jedna proměnná, konstanta nebo funkce v modelu je náhodná veličina nebo náhodná funkce.

Fuzzy modely Některé proměnné, konstanty nebo funkce jsou fuzzy veličiny, nebo fuzzy funkce.

Podle povahy problému se modely používají individuálně nebo v kombinacích. Pro řešení známých problémů lze použít tzv. standardní modely. Pro řešení nových problémů je třeba konstruovat nové modely.

1.3 Modelování neurčitosti, nejistoty a rizika

Nejistotou při zobrazení systému pomocí matematického modelu rozumíme situaci, kdy nemáme k dispozici všechnu potřebnou informaci nebo kdy některé z informací jsou nespolehlivé.

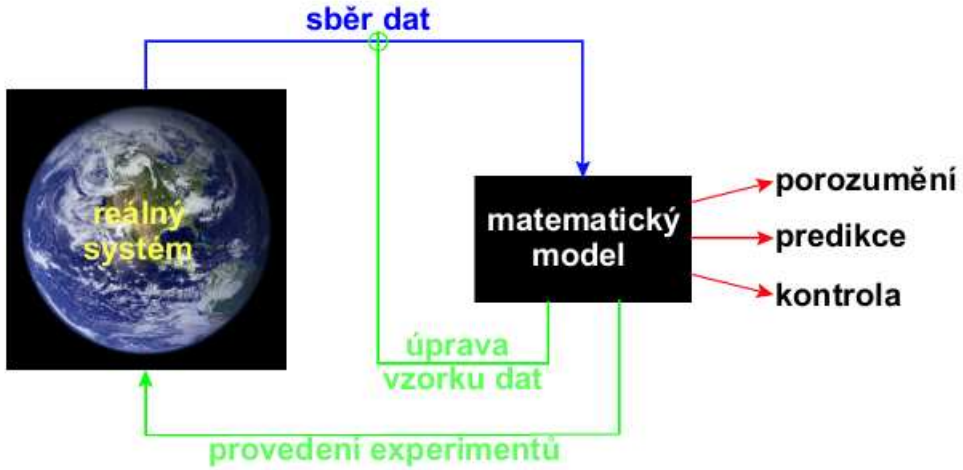
Modelování při riziku předpokládá, že některé informace jsou náhodné veličiny, nebo že některé procesy jsou popsány náhodnými funkcemi. V případě modelů s rizikem můžeme velikost rizika při přijetí řešení popsat pomocí pravděpodobnostních charakteristik.

Analogicky můžeme považovat modelování za rizika i v případě použití fuzzy veličin, nebo fuzzy funkcí. Velikost rizika lze potom vyjádřit buď pomocí vhodné fuzzy míry nebo tuto fuzzy míru transformovat na subjektivní pravděpodobnost.

Kapitola 2

Metodologie matematického modelování

Na obrázku 2.1 je znázorněn proces zkoumání systému, sběr z toho vyplývajících dat, jejich tok do matematického modelu a na základě analýzy modelu (pochopení, předvídání a úprava), přizpůsobení sběru dat včetně provádění experimentů na systému.



Obrázek 2.1: Proces zkoumání systému s využitím matematického modelování

[1]

2.1 Obecné zásady matematického modelování

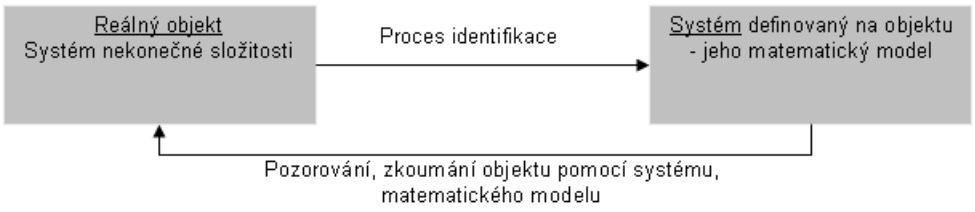
Matematické modelování je odborná a kvalifikovaná činnost, vyžadující týmovou spolupráci odborníků z různých oblastí: odborníka z oblasti oboru řešené problematiky, specialistu v oblasti matematiky, specialistu z oblasti informatiky, apod. Pro úspěšné matematické modelování musí být splněny následující předpoklady:

[1] zdroj: http://math.ucsd.edu/ dmeyer/

- Znalost metod a prostředků matematické a funkcionální analýzy, algebry a diskrétní matematiky. Je důležitá při volbě správné metody řešení a modelu.
- Znalost techniky modelování a zdrojů informací o modelu. Úsilí vynaložené na konstrukci a využití určitého modelu z literatury musí být úměrné jeho přínosu.
- Týmová spolupráce. Musí existovat dostatečný prostor pro vlastní vývoj matematického modelu (iniciativa) a musí být zainteresovanost (studijní, výzkumná) na využití modelové techniky (motivace).
- Výpočetní základna. Všechny tři složky ICT, tj. hardware, software a komunikace musí být řešitelskému týmu k dispozici a musí být v rovnováze.
- Informační a datová základna. Každý model je třeba ověřit pomocí vstupních parametrů a dat, které vycházejí z konkrétních hodnověrných údajů a zdůvodněných odhadů. Údaje musí být ve formě vhodné pro ověřování modelu. Je třeba vytvářet specifické informační systémy (banky dat), které uchovávají data.
- Experimentální základna. U složitých matematickým modelů by měla být k dispozici i experimentální základna, kde se na ověří v praxi řešení modelu.

Metodologie matematického modelování se vyvíjí jako samostatná odborná specializace [10], která se v současné době zařazuje do Teorie modelování jako součást tzv. Systémové analýzy. Obecné zásady, které je třeba při matematickém modelování systémů respektovat, lze velmi zjednodušeně popsat následujícími kroky:

1. Identifikace systému z hlediska matematického modelování.
   - Rozhodnutí, zda se jedná o standardní systém (problém), již řešený a volba standardního modelu.
   - Rozhodnutí, zda se jedná o nový, dosud neznámý systém a zda použijeme upravený standardní model nebo vytvoříme model nový. K tomu je třeba zpravidla vytvořit tvůrčí odborný tým.
   - Rozhodnutí, zda model bude statický, dynamický, dynamizovaný, deterministický, stochastický. Zda bude deskriptivní, nebo normativní. Zda systém bude modelován jedním modelem či více modely a jak budou vzájemně uspořádány (propojeny).



Obrázek 2.2: Identifikace systému

2. Konstrukce modelu systému.

   - Prvky systému: elementární část systému při dané rozlišovací úrovni dále nedělitelná
   - Vazby: vzájemné závislosti mezi prvky (kauzální vztahy – příčina x následek, způsoby spojení mezi prvky, souvislosti mezi jevy, informační vazby, matematicky formulované vztahy, atd.)
   - Struktura systému
   - Chování systému
   - Okolí systému: množina prvků, které nejsou prvky systému, ale mají k němu významné vazby
   - Organizace dat v systému: jejich struktura, způsobu uložení, vstupu a výstupu dat, atd.
   - Validita modelu systému: Ověření matematických předpokladů, stability, konsistence a konvergence řešení, atd.

3. Výpočet řešení modelu.

   - Volba algoritmu řešení.
   - Výběr variant řešení.

4. Výběr užší skupiny dostatečně dobrých řešení.

   - Výběr vhodných řešení se provádí v rámci algoritmu řešení.
   - Výběr vhodných řešení provádí specialista z oboru řešené problematiky.
   - Výběr vhodných řešení provádí skupina expertů.

5. Experimentování s vybraným řešením.

   - „What-if“ analýza[2], „Goal seeking“ problém[3].
   - Scénáře.

6. Výběr optimálního řešení.

7. Implementace.

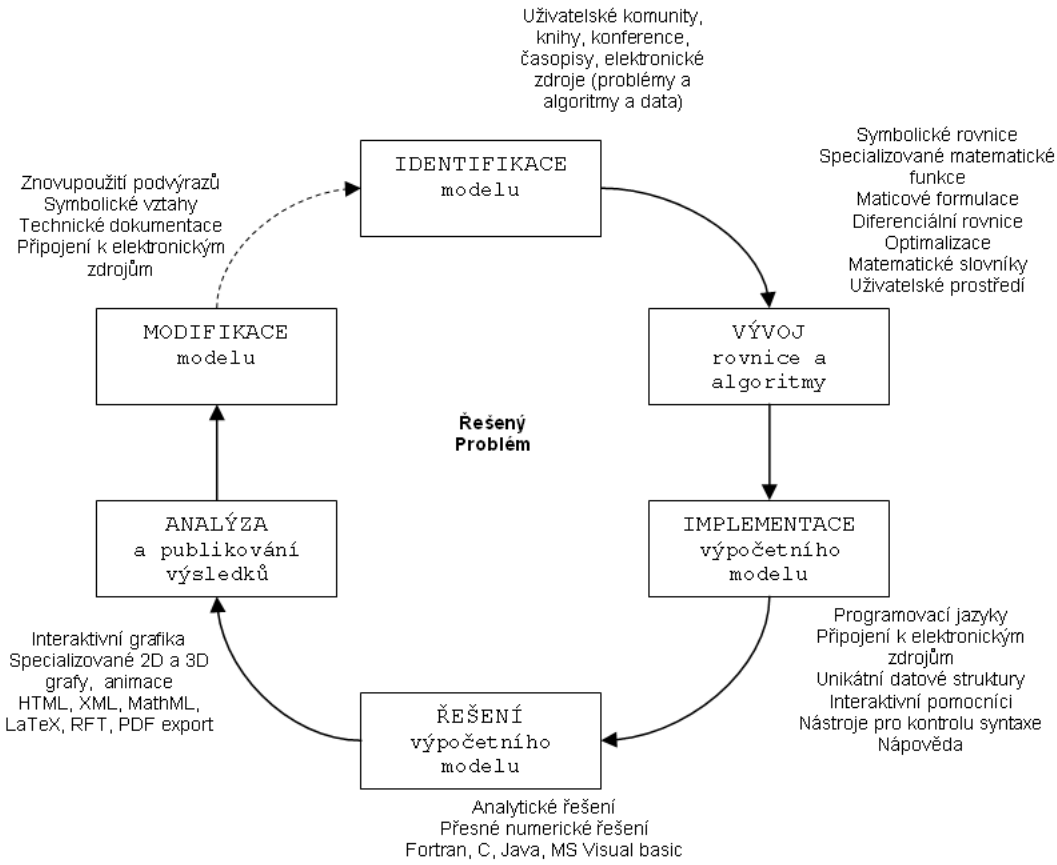
   - Monitoring implementace.
   - Sledování zpětné vazby.
   - Úpravy modelu a nová implementace.

[2]česky řečeno „Co se stane když“ analýza se používá většinou při numerické simulaci, která slouží k tomu, abychom odhalili jak je model citlivý, na odhad vstupních parametrů, které by měly ležet v nějakém vhodném intervalu.

[3]souvisí s tím, že v praxi se vytváří matematický model metodou postupných kroků lépe aproximující řešený systém. Toho se dosahuje lokálním hledáním dostatečně přesného řešení. Tento problém je v anglické literatuře nazýván jako „satisfying problem“, „feasibility problem“, nebo také „goal-seeking problem“.

2.2 Matematické modelování s využitím ICT

Postup při matematickém modelování reálného problému s ICT je uveden na obrázku 2.3 a sestává z několika kroků (Hřebíček, 2006). Jde o permanentní interaktivní proces s četnými zpětnými vazbami, který se několikrát opakuje.



Obrázek 2.3: Matematické modelování s využitím ICT

Tento postup vychází z postupů v projektovém řízení. Nejprve je nutno si vyjasnit cíle, které chceme dosáhnout. Ty určí směr řešení ve dvou směrech:

Na jaké úrovni podrobnosti chceme zkoumat objekt.

Např. při modelování populačního růstu k napomáhání pro zemědělské poradce, empirický model obsahující podmínky pro nejdůležitější faktory růstu může být naprosto dostatečný. Model může být považován jako souhrn veškerého současného vědění. Tento model je jednoznačným výzkumným nástrojem pro velmi limitované použití jako například pro návrh experimentů ke studiu procesu výživy přežvýkavců.

Za druhé musíme vytvořit hranici mezi modelovaným systémem a jeho přirozeným prostředím. Toto rozdělení je správné, pokud prostředí ovlivňuje chování systému, ale modelovaný systém neovlivňuje toto prostředí.

Např. při modelování růstu malé kolonie jehličnanů k prognóze výnosu produkce dřeva je vhodné zahrnout počasí jako část prostředí. Vliv počasí na růst může být začleněný využitím statistických klimatických dat z podobných lokalit a několika uplynulých letech. Avšak žádný růstový model světových lesů nemůže obsahovat ovlivnění počasí skrze vývoj tohoto společenství, přestože je znám podstatný vliv lesního porostu na obsah CO_2 v atmosféře.

2.2.1 Identifikace modelu

Identifikace (stanovení) jednotlivých složek matematického modelu s využitím odborné literatury (knihy, časopisy, ..), spoluprácí s odbornou a vědeckou komunitou a dále s využitím ICT k vyhledání a sdílení znalostí o řešeném problému je prvním krokem, který musíme udělat při vytváření matematického modelu. Správná matematická formulace zkoumaného problému je velmi důležitá pro další postup řešení.

Je třeba vyjít z analýzy systému, z celkového jeho chování a stanovených cílů řešení. Realita je složitá, je třeba ji vymezit a pro účely modelu zjednodušit. Proto definujeme v rámci objektivní reality systém, tj. prvky, vazby, vstupy a výstupy, procesy a funkce. Dále provádíme zjednodušení (simplifikaci) problému, kdy nepodstatné oddělujeme od podstatného.

Tvorba předpokladů

Když jsme rozhodli o modelovaném systému, musíme vytvořit základní strukturu modelu, která musí reflektovat naše předpoklady a domněnky o tom, jak systém funguje [11]. Tyto domněnky mohou být uvedeny do formy základních předpokladů. Budoucí analýzy systému vždy zachází s těmito předpoklady jako s pravdivými, ale výsledky těchto analýz budou validní pouze pokud tyto předpoklady jsou platné.

Newton tedy předpokládal, že hmotnost je obecně konstantní, kdežto Einstein uvažoval hmotnost jako proměnlivou. To je jeden z elementárních rozdílů mezi klasickou mechanikou a teorií relativity. Aplikací výsledků klasické mechaniky na objekt pohybující se rychlostí blízkou rychlosti světla vede k nesrovnalostem mezi teorií a pozorováním. Pokud jsou předpoklady dostatečne precizní mohou okamžitě vést přímo k matematickým rovnicím popisujícím modelovaný systém.

Příklad Při studiu růstu populací je běžným předpokladem, při absenci limitujících faktorů, růst populace úměrný její velikosti. Deterministický model popisující průběh růstu této populace v čase je zadán diferenciální rovnicí:

$$\frac{dp}{dt} = ap$$

kde $p(t)$ je velikost populace v čase t a a je konstanta. Řešením této rovnice integrací dostáváme:

$$p(t) = p(0)e^{at}$$

kde $p(0)$ je velikost populace v nulovém čase (na počátku). Vzhledem k řešení je vidět, že populace roste exponenciálně.

Je zřejmé, že ne všechny populace rostou exponenciálně rychle. Protože diferenciální rovnici jsme odvodili z předpokladů, musíme se zaměřit na předpoklady kvůli vysvětlení této odchylky.

V tomto případě je vysvětlením absence limitujících faktorů. Mnoho přírodních populací je omezeno různými okolnostmi jako zdroj potravy, lokalitou výskytu – velikostí nebo možnostmi tuto oblast uhájit proti nepřátelům, což omezuje možnosti růstu populace.

Dalším důležitým požadavkem je aby všechny předpoklady byly určeny jednoznačně a výstižně. To nám umožní vrátit se k nim později a zhodnotit jejich náležitosti. Další předpoklad, který použijeme při vytváření diferenciální rovnice, je rovnoměrné zabírání místa při růstu populace. Pokud se populace skládá z oddělených generací, můžeme použít diferenční rovnici:

$$d_{i+1} = bd_i,$$

kde d_i je velikost i-té generace. Tato rovnice má řešení:

$$d_i = d_0 b^i,$$

kde d_0 je počáteční velikost populace. Poznamejme, že řešení diferenciální a diferenční rovnice může splývat v čase $t = i$ právě když $d_0 = p(0)$ a $b = e^a$.

2.2.2 Sestavení modelu

Sestavení modelu (vývoj matematických rovnic a formulí) včetně matematické analýzy (korektnost, konsistence, stabilita a konvergence řešení) [5] je dalším důležitým krokem v matematickém modelování. Pro konstrukci modelu je rozhodující účel, který sledujeme. Ten rozhoduje o tom, co budeme ve skutečnosti pokládat za významné a co zahrneme do modelu a co jako podružné ponecháme mimo model a mimo naše úvahy. Důležitá je zde analýza citlivosti jednotlivých parametrů modelu a minimalizace jeho neurčitosti. Tvorba modelů patří k tvůrčí činnosti a vyjadřuje kromě dobré znalosti modelové techniky a také dobrou znalost věcné problematiky. Každý model musí vycházet z konkrétní hypotézy odvozené z objektivní reality (skutečnosti).

Výběr matematických rovnic

Po té, co bylo rozhodnuto o struktuře modelu, musí být vybrány matematické rovnice k popsání modelovaného systému. Je velmi rozumné vybrat tyto rovnice pečlivě, protože mohou nepředvídatelně ovlivnit chování matematického modelu.

Matematické rovnice z literatury

Může se stát, že někdo další publikoval matematické rovnice týkající se problému o který se zajímáte. To poskytuje dobrý výchozí bod, ale je nezbytné postupovat s těmito informacemi obezřetně. Problémy se kterými se můžete setkat mohou být následující:

- Rovnice jsou odvozené z dat v oblasti vysvětlujících proměnných, která neobsahuje oblast nutnou pro aplikaci modelu,
- Experimentální podmínky (prostředí) se podstatně liší od podmínek vyskytujících se v našem modelovaném problému,

- Rovnice popisují chování většiny dat aniž by uvažovaly známé odchylky na koncích oblasti dat, nebo jejich variabilitu (proměnlivost).

Některé oblasti vědy jsou dostatečně dobře prostudované, a tak odpovídající formulace analýz se staly standardem. Proto je relativně bezpečné uvažovat podobné analýzy (a proto i struktury rovnic) za řešení podobných problémů.

Často nejsou rovnice v literatuře vyjádřeny v přesné formulaci pro hledaný model. Pojmenované a pomocné proměnné mohou být přesunuty během regrese. Rovnice také může popisovat změnu váhy zvířat vzhledem k času, přestože model potřebuje znát změnu počtu jedinců v čase. V jiném případě můžeme přijmout parametr pouze odhadující přibližnou hodnotu, protože není nejdůležitějším pro naše potřeby.

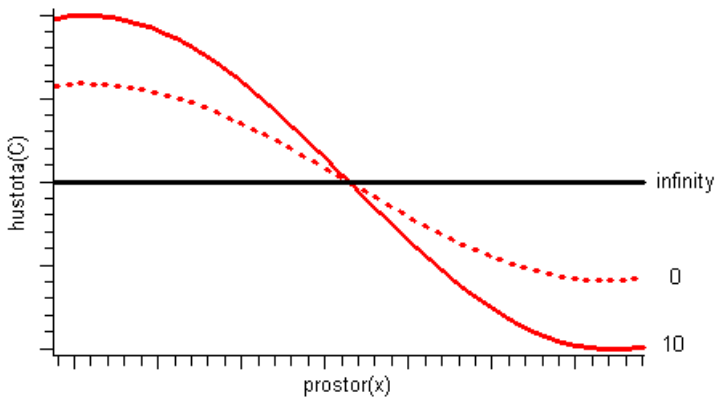
Analogie z fyziky

Fyzici vybudovali mnoho matematických modelů k popsání široké palety fyzikálních systémů. Obvykle jsou tyto systémy specifikovány velmi precizně a relativně usnadňují použití matematických rovnic.

Existuje mnoho případů, kdy můžeme využít tyto znalosti při návrhu rovnic pro biologické systémy. Například lokální migrace organismů je často považována za ekvivalentní procesu difúze. Difúze velkého počtu malých částic je velmi dobře prostudována a je popsána rovnicí:

$$\frac{\partial C(x,t)}{\partial t} = D\frac{\partial^2 C(x,t)}{\partial x^2},$$

kde $C(x,t)$ je koncentrace částic v lokalitě x a čase t. Tato rovnice nedává žádný odhad budoucího pohybu jednotlivých částic a je pouhým jednoduchým popisem kolektivního chování. Obrázek 2.4 ilustruje jak se mění prostorové rozšíření populace v závislosti na čase podle rovnice difúze.



Obrázek 2.4: Difúze populace, ve které nejsou žádné přírůstky ani úmrtí

Zkoumání dat

Pokud neexistují žádné informace o matematických vztazích, jedinou cestou v řešení je získání tvaru dat a přizpůsobení rovnic těmto datům [12]. To má výhodu pro další kontrolu při analýze. Když naše data se sestávají z měření y_i v čase t_i, $i = 1..n$, můžeme odhadovat rozdíl mezi časy t_i a t_{i+1} pomocí $d_{i+1} = \frac{y_{i+1}-y_i}{t_{i+1}-t_i}$. Vykreslením d_{i+1} proti $\frac{y_{i+1}+y_i}{2}$ získáme odhad průměrné hodnoty y. Dále můžeme vyšetřit vztah mezi $\frac{dy}{dt}$ a y. V případě exponenciálního růstu můžeme očekávat graf zobrazující rovnou přímku. Pokud je růst omezen, můžeme očekávat nějakou odchylku od linearity. Například logistický růst [13] popsaný rovnicí

$$\frac{dy}{dt} = ry(a - y),$$

vede ke kvadratické křivce. Protože je poněkud složitější vizuálně rozhodnout zda je či není křivka kvadratická, chceme nalézt graf, který nám zobrazuje rovnou přímku, pokud velikost růstu sleduje logistickou rovnici. To nás motivuje k zápisu logistické rovnice následovně:

$$\frac{1}{y}\frac{dy}{dt} = ry(a - y),$$

Pokud vyřešíme levou stranu rovnice, někdy nazývanou jako poměrný růstový koeficient, z dat, vykreslení grafu proti y opravdu dá přímou linku. Zde je matematický výsledek, který jsme přávě formulovali:

$$\frac{1}{y}\frac{dy}{dt} = \frac{d}{dt(log(y))},$$

Proto pokud vypočítáme $d_i = \frac{log(y_{i+1})-log(y_i)}{t_{i+1}-t_i}$ můžeme pokračovat jako výše. Postup výpočtu je demonstrován na obrázku 2.5. Odchylka od linearity ve čtvrtém grafu nejen indikuje to, že vytvořená logistická křivka není korektním modelem dat, ale též může navrhnout typ odpovídající úpravy.

2.2.3 Implementace modelu

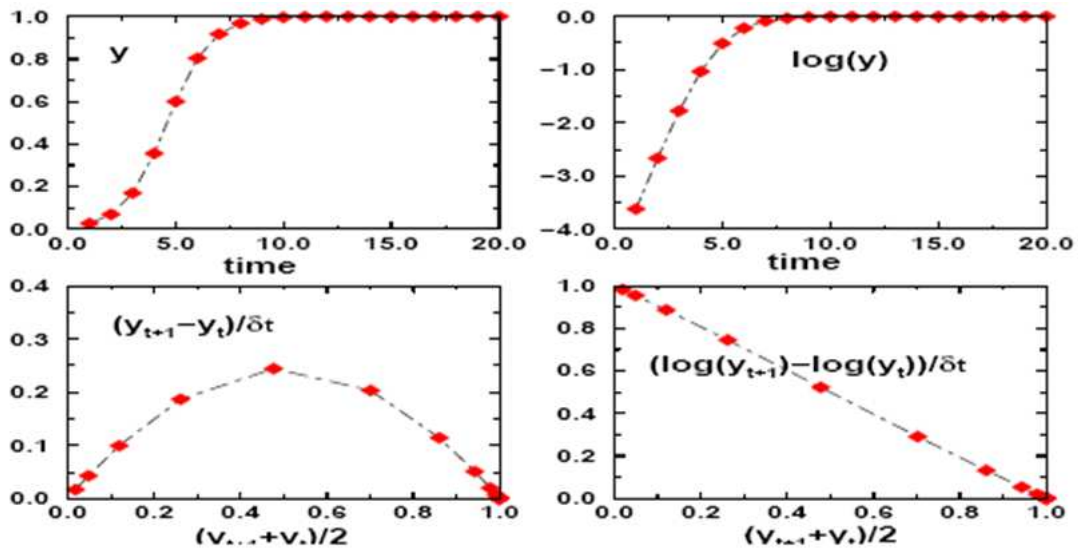
Implementace modelu s využitím ICT (naprogramování v příslušném programovacím jazyce, jeho odladění a verifikace, analýza jeho výpočetní složitosti a využití příslušného hardware, atd.). Zde se vyplatí využít moderních ladících technik, případně i použít již vyvinuté a dostupné (open source) knihovny, služby i moduly z Internetu.

2.2.4 Řešení modelu

Řešení implementovaného modelu s využitím ICT nástrojů (analytické, numerické, atd.). Naplnění modelu konkrétními parametry a daty. Je třeba dbát na jejich hodnověrnost. Existují dva způsoby odvození řešení z modelu:

Analytické (explicitní) řešení spočívá v nalezení přesného řešení pomocí analytických matematických metod (řešení soustav rovnic, řešení úlohy na vázaný extrém apod.).

Numerické (přibližné) řešení se používá při řešení modelů, u kterých neumíme problém řešit analyticky, nebo v případech, kdy je analytické řešení obtížné a složité (metody



Obrázek 2.5: Relace mezi logistickou křivkou a daty populace

Monte Carlo, simulace na počítači apod.). Při numerickém řešení musíme uvažovat jeho numerickou stabilitu a konvergenci a chybu, která nám vznikne.

2.2.5 Analýza řešení modelu

Verifikace řešení (kontrola zda výsledky souhlasí s chováním objektu), jeho vizualizace atd. a publikace výsledků. Model je jen přibližným obrazem objektivní reality. Je dobrý, jestliže umožní přesně sledovat důsledky změn ve vstupech do systému na výslednou efektivnost systému. Cílem testování modelu je prověření jeho správné struktury, vypovídací schopnosti, formálních kvantitativních vlastností včetně odstranění formálních chyb. Testování modelu provádíme tak, že modely naplníme empirickými číselnými údaji, dosažené výsledky analyzujeme a porovnáváme s realitou. Ověřování lze promítat i do minulosti („ex post“) i do budoucnosti („ex ante“). Interpretační analýza představuje převod výsledků do reálného systému. Je to aktivní proces, při kterém je třeba provádět neustále logickou kontrolu smyslu řešení, vyhnout se nebezpečí mechanického používání modelové techniky. Významným prvkem interpretace je promítnutí výchozích hypotéz a předpokladů do výsledku řešení. Shrnutí získaných poznatků včetně všech aspektů, které nebyly do matematického modelu zahrnuty.

Dalším důležitým požadavkem je aby chování modelu mohlo být popsáno dvěma způsoby. Kvalitativní popis dává odpověď na otázku „jak?“, zatímco kvantitativní popis dává odpověď na otázku „jak moc?“. Obecně kvalitativní chování bude vždy stejné pro všechny modelované skupiny a z toho důvodu podléhá základním výsledkům. Toto omezení značně oponuje kvantitativnímu chováním, které je často vázáno k jednotlivým okolnostem.

Kvalitativní chování stochastických modelů je pravděpodobně lepší pro ukázání větší rozmanitosti než příslušné deterministické modely. Například různé realizace stochastického populačního modelu mohou projevovat exponenciální růst a vyhynutí. Se stochas-

tickým modelem je tedy důležité popsat nejen průměrné chování, ale také rozmezí typů chování.

Analýza citlivosti

Cílem analýzy citlivosti [6, 14] je měnit parametry modelu a vyhodnotit tyto změny ve výsledcích modelu. Tato metod je obzvláště užitečná k identifikování slabých míst modelu. Tato místa pak mohou být zvýrazněna experimentováním nebo jednoduše popsána a vyznačena pro opatrné zacházení při aplikaci modelu.

Pokud je model obzvláště jednoduchý, je možné rozdělit výsledky řešení pro každý parametr v několika krocích. Odvozením získáme přesné poměry změn prognózy vázané k jednotlivým parametrům. Ve složitějších modelech je lepší se dělení úplně vyhnout a použít některé numerické metody.

Jak rozhodnout o velikosti změny každého parametru? Tato změna může záviset na tom, jak přesnou hodnotu parametru chceme získat. Z toho důvodu parametry, které jsou odvozeny ze vstupních dat, mohou obsahovat mnoho standardních chyb. Některé parametry mohou být odhadnuty. V tom případě je nutné mít procentuální odhad spolehlivosti tohoto odhadnutého parametru. Měli bychom stále pamatovat, aby člověk odhadující parametr, příliš nepřecenil jeho spolehlivost. Obzvláště bychom měli dávat pozor na parametry ve vzájemném vztahu, protože změna jednoho z nich může vést k případné změně dalších.

Metody testování

Po té, co jsme dokončili návrh modelu a jsme s ním spokojeni, přichází čas k otestování modelu proti měřením z fyzického systému, který reprezentuje. Tento proces je obvykle nazýván validací modelu. Podobné slovo, verifikace, je obecně využíváno pro netriviální problém kontroly, zda jsou prognózy získané modelem přesné.

Jaká data můžeme využít při testování modelu? Zde je kladen velký důraz na použití jiných dat, než které jsme využili při odhadu parametrů modelu. Využití těchto dat by nás mohlo mylně dovést k závěru, že model dává mnohem lepší výsledky než by reálně vykazoval.

V lineární regresi opravujeme účinek odhadu dvou parametrů vydělením součtu čtverců odchylek hodnotou (n-2) místo (n). V komplikovanějších případech nemáme žádný skutečný ekvivalent počtu volnosti argumentu, takže musíme postupovat s větší opatrností.

Prognóza pomocí dříve nepoužitých dat

Nejpřesvědčivější metodou testování odhadnutých parametrů modelu je použití dat, která nemají žádnou souvislost s předchozím vývojem. Použitím této metody, zredukujeme potenciální možnost výskytu chyb a získání neopodstatněně dobrých shod mezi modelem a realnými daty.

Fráze „nemající žádnou souvisloust“ je velmi důležitá. Předpokládejme, že jsme parametry modelu odhadly z dat posbíraných během několika let na jedné farmě. K otestování parametrů však použijeme data získaná z následujících let na téže farmě. Co nám to může říct o daném modelu? Při nejlepším můžeme říct, že vytvořený model velmi dobře předpovídá vývoj na této jedné farmě. Rozhodnutí, zda model můžeme použít na širokém spektru různých farem, musí být ověřeno na reprezentativním vzorku různých farem.

Jaké statistické metody lze použít při hledání nesrovnalostí mezi daty a prognózou modelu?

Pro prognózy P_i a měření $O_i, i = 1..m$, můžeme použít následující statistické metody:

Průměrná chyba (Bias zn. B)

$$B = \frac{1}{n}\sum_{i=1}^{n}(P_i - O_i)$$

Směrodatná odchylka (Standard Deviation zn. SD)

$$SD = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(P_i - O_i - B)^2}$$

Střední kvadratická chyba (Mean Square Error zn. MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(P_i - O_i)^2 = SD^2 + B^2$$

Každá z těchto rovnic může být vynesena do grafu vůči vybrané proměnné k testování homogenity. Často bývá dobrou myšlenkou vyvážit tyto sumární statistické výsledky průměrem z pozorování. To umožní vytvořit nezávislost na jednotkách měření.

Intuitivně vypadá, že graf měření oproti prognóze by měla být jednotková přímka procházející počátkem pod úhlem 45 stupňů. Přestože je zajisté pravda, že od dobrého modelu očekáváme rovnou přímku, není pravdou, že očekávaná regrese je jednotková přímka. Důvod tohoto zjevného paradoxu je obtížné pochopit, ale přesto je to výhoda, kterou snadno vypozorujeme.

Další pokročilé statistické metody lze dohledat v [12, 15]

Důvody pro odhad chyb

Jaké jsou důvody nedokonalých prognóz? Pokud pochopíme, kde se během prognózy stala chyba, potom můžeme rozhodnout jak na ni reagovat. Dle obecných zásad existují tři důvody vzniku chyb:

- Prvním je přirozená variabilita systému a jeho prostředí. To je to, co běžně považujeme za chybu měření v našich experimentech. Tyto chyby můžeme sledovat a odhadovat jejich dosah z předchozích prací a případně se jich vyvarovat. V těchto případech víme, kam se podívat po dalších zdrojích informací o těchto chybách.

- Druhým je vliv faktorů, které jsme zanedbali. Jestliže mohou být tyto faktory rozeznány od přirozené variability systému závisí na mnoha dalších informacích, dodatečně získaných o tomto problému. Je přínosné strávit čas uvažováním jak tyto nepříjemné vlivy omezit, například sledováním pomocí grafických metod, protože změna struktury modelu může být již velmi časově a kapacitně náročná.

- Třetí důvod vzniku chyb je způsoben chybami v návrhu modelu. To může být způsobeno špatnou analýzou nebo nevhodně zvolenými parametry. V tomto případě musíme provést úpravu rovnic modelu, aby lépe reflektoval testovací data. Stále ale musíme uvažovat data, která jsme použili při vývoji modelu, aby tyto změny byly stále kompatibilní.

Odhadování parametrů modelu

Odhad parametrů modelu [13, 16] zjevně předchází vyhodnocení souhrnu vlastností modelu, avšak diskutujeme ho právě zde, jelikož souvisí s měřením souhrnu vlastností modelu (B, SD a MSE) diskutovanými výše.

Jakmile máme množinu experimentálních dat D a chceme určit parametry modelu z těchto naměřených dat, je obvyklým postupem minimalizovat jeden z našich testovacích výpočtů s ohledem na hodnoty parametrů. To nám umožní získat nejlépe souhlasící množinu parametrů. To nám také pomůže vysvětlit, proč používat data z různých měření pro odhad a testování modelu, aby odhad parametrů měl dobré výsledky i při dalším použití. Běžnou praxí pro odhad parametru je využití minimalizace střední kvadratické chyby. Výhodou této metody, je možnost uvažovat chyby v datech jako normálně distribuované a nekorelované mezi měřeními O_i, pak sklon povrchu chyby okolo minima hodnot parametru může být použit při výpočtu standardní chyby v odhadu parametru. Problémem se pak stává, že předpoklad normality nemusí být platný a pak data často nemůžeme uvažovat jako nekorelovaná (například velikost populace nebo jakoukoliv jinou proměnnou) v čase.

V případě stochastických modelu existují lepší statistické metody k odhadu parametrů od doby, kdy je možné zkonstruovat pravděpodobnost $L(D|p)$, což je jednoduchá pravděpodobnost, že model využívající parametry p vygeneruje pozorovaná data D. Pravděpodobnost je funkcí parametrů modelu (a pozorovaných dat) a může proto být považována jako pravděpodobnost parametrů daná daty.

Maximalizování vzhledem k parametrům můžeme obdržet maximální pravděpodobnost odhadu parametrů a opravdu také známe jejich plnou pravděpodobnostní distribuci (včetně standardních chyb). Navíc jsme nevytvořili žádné další předpoklady. Kde je tedy chyták? Naneštěstí výpočet pravděpodobnosti je velmi obtížný, pokud data jsou ovlivněna nějakou událostí (jako narození či smrt). V těchto případech musíme chybějící data zjišťovat typicky nějakou průměrovací metodou jakou jsou Markovovy řetězce Monte Carlo (MCMC). Tato oblast se rapidně vyvijí, ale mnoho parametrů bývá odhadováno pomocí minimalizace střední kvadratické chyby.

Porovnání dvou modelů pro stejný systém

Pokud jsou dostupné dva modely pro stejný systém, můžeme chtít tyto modely porovnat, abychom vybrali, který z nich budeme v budoucnosti používat. Toto porovnání bude vždy záviset na jistém prvku subjektivnosti rozhodovatele, protože se zde vyskytuje mnoho různých aspektů, na kterých může být rozhodnutí založeno.

Příkladem těchto aspektů může být obecnost, možnost predikce[4], výpočetní prostředí,

[4] v kontextu vědeckých výpočtů je predikce důsledné (často kvantitavní) prognostické tvrzení, co se stane za určitých podmínek, typicky vyjádřeno ve formě „Pokud je A pravda, pak také B je pravda“. Vědecká metoda je založena na testování tvrzení, které je logickou sekvencí vědeckých teorií. To je vykonáno díky opakovaným experimentům nebo pozorovacími studiemi.

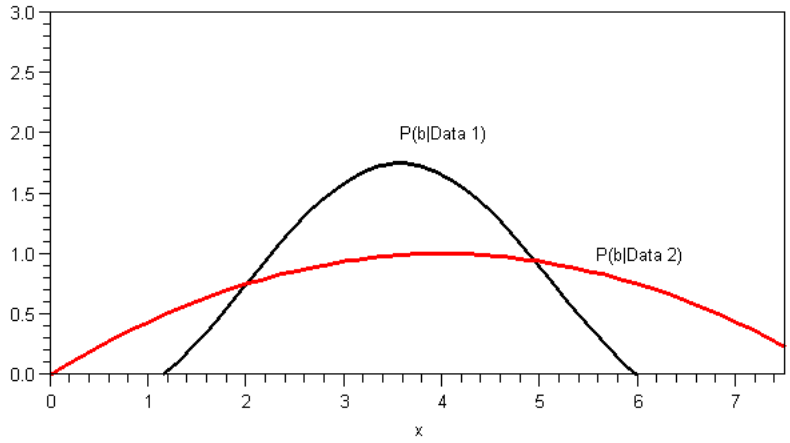
provedené systematické analýzy, zvážení vytvoření, nastudování a testování modelu.

Dobrým počátečním místem pro rozhodování jsou základní předpoklady modelu. Jak se liší? Patrně zde existují nějaké rozdíly, jinak by modely byly identické. Pokud jsou rozdíly ve struktuře modelů, je pravděpodobné, že byly navrženy pro použití v různých situacích. Který se více blíží předpokládáné aplikaci? Jak důležité jsou chybějící vzájemná působení, o kterých víme?

Po té, co jsme analyzovali rozdíly, jsme v pozici, kdy musíme rozhodnout jak moc ovlivňují budoucí aplikaci modelu. Zmapujeme predikce modelů při využití velkého rozsahu pečlivě vybraných scénářů. Liší se významně? Dva pohledy zde mohou být stresující. Prvním pohledem je, že malé rozdíly jsou běžné a všechny modely jsou pouze aproximativní. Druhý varuje před modely, kde jsou výpočty prováděny iterativně, protože se malé odchylky mohou zjevně kumulovat do podstatné chyby.

Pokud jsou všechny výsledky rozhodování ekvivalentní, musí být finální rozhodnutí věcné. Zkuste porovnat predikce všech modelů na nezávislých pozorováních. Porovnejte modely pomocí statistických metod. V tento čas při testování modelů lze též otestovat na všech modelech potenciální aplikaci modelu na adekvátních datech.

Obrázek 2.6 ukazuje příklad, kde jsou parametry modelu odhadnuty z dat ze dvou vyšetření. V prvním vyšetření byl parametr b odhadnut s vysokou přesností. Avšak v druhém vyšetření byl tento parametr velmi špatně odhadnut (v důsledku velké chyby). Z toho vyplývá, aby druhý dataset nebyl použit v modelu a proto alternativní model bez této komponenty bude v tomto případě preferován.



Obrázek 2.6: Porovnání dvou modelů s různou přesností odhadu parametrů

2.2.6 Modifikace modelu

Modifikace modelu a jeho vylepšení. V případě, že dosažené řešení není v dostatečném souladu s objektivní realitou je nutno začít znovu postupovat od kroku 1 a opakovat celý předešlý postup matematického modelování do té doby dokud nedosáhneme uspokojivého řešení zkoumaného problému.

Metody prezentace modelu jeho potenciálním uživatelům závisí na znalostech uživatele modelu a matematického modelování obecně. Pokud chce uživatel vědět raději méně o detailech modelu, je vhodné ukázat mu všechny relevantní informace o výstupech modelu. To umožní uživateli (který není programátorem) vytvořit si objektivnější pohled na řešení modelu a jeho interpretaci. Je dobré zkontrolovat zda predikce řešení zahrnují skryté extrapolace. Tyto extrapolace mohou hrát úlohu buď vzhledem k použitým datům při vytváření modelu nebo vzhledem k datům použitým při testování modelu.

Predikce s odhadem přesnosti

Jestliže jediným výstupem z modelu je predikce nějaké kvantity, jak může uživatel odhadnout přesnost predikce? Bohužel to není možné a uživateli nezbude nic jiného než výsledku důvěřovat nebo ne. Proto by bylo lepší aby predikce byla doprovázena odhadem přesnosti, jako je velikost směrodatné odchylky nebo interval spolehlivosti. Tyto údaje mohou být získány již při studiu modelu nebo při jeho testování.

Pokud jsme vyšetřili efekt chyby v odhadnutých parametrech již během nastudování modelu, můžeme odhadnout i přesnost predikce z jednoduché sumarizace distribuce potenciálních výsledků. Pokud model neobsahuje velké množství nesprávných vztahů, tak nám to umožní minimalizovat výslednou chybu modelu.

Eventuelně odhad chyby může být převzat z predikce chyby vytvořené během testování modelu. To je nejlepší možná varianta, protože během testování bývá využito mnoha měření z různých zdrojů. Přímá aplikace odhadu chyb predikce je kalkulace míry spolehlivosti příslušných vztahů.

Příklad Z experimentů [17] víme, že krmíme-li zvířata stejnou potravou, rostou mírně odlišnou rychlostí. Proto libovolný model popisující růst skupiny jedinců musí obsahovat náhodný prvek, který bude reprezentovat právě tuto odlišnost. Pokud je cílem, aby každé zvíře získalo denně jeden kilogram váhy a zjistíme, že průměrný růst je právě 1 kg denně, polovička zvířat tohoto cíle ani nedosáhne. Abychom získali alespoň 95% úspěšnost potřebujeme nastavit průměr na hodnotu 1 kg denně + 1,6 · směrodatná odchylka (uvažující normální distribuci).

Příklad: Podpora rozhodování Uvažujme úlohu rozšíření ekonomického modelu o podporu rozhodování [2]. Náklady jsou připojeny do různých vstupů modelu jako například zvířecí potrava nebo rostlinné hnojivo. Vstupní úrovně jsou vybrány tak aby splňovaly všechna omezení systému.

Podmínkou pro úrovně těchto vstupních dat je použití odhadnutých biologických výstupů modelem. Samotné biologické výstupy mají připojené finanční hodnoty (např. prodejní ceny). Rozdíl mezi výslednými a vstupními hodnotami jsou odhadem hrubého zisku.

Cílem ekonomických analýz je najít strategii, která bude nejvíce výhodná a zisková. Pro jednoduché modely a několika málo proměnnými není obvykle obtížné tento problém vyřešit. Pokud jsou v modelu zahrnuty pouze dvě proměnné, lze pro analýzu použít jednoduché grafické zpracování. Numerické metody jako je lineární nebo kvadratické programování jsou použitelné, ale nepřesnosti v predikci modelu nebo ekonomických podmínkách mohou generovat navíc další významná čísla spočítaná špatně.

Pro komplexní systémy s mnoha proměnnými může být optimalizace velmi obtížný problém. Ve speciálních případech je možné přizpůsobit daný problém pro řešení pomocí lineárního nebo kvadratického programování. Alternativně lze použít metodu *Response surface metodology*, kde za výslednou proměnnou predikce vezmeme zisk. Mohou být též využity pokročilé metody jako dynamické programování, avšak jejich použití již není příliš triviální.

Pokud jsou výsledky modelu náhodné, vytvořit tvrzení jako „strategie A je vždy ziskovější než strategie B“ bývá možné jen ve zcela ojedinělých případech. Místo toho tvoříme tvrzení o pravděpodobnostech: „průměrný zisk při použití strategie A je větší než průměrný zisk získaný strategií B“. Jestliže je toto tvrzení pravdivé, je vždy lepší vybrat strategii A? Odpověď je ne. Může se stát, že strategie A má 40% šanci nás dovést do bankrotu, ale 60% šanci vytvořit nám obrovský zisk. Ne každý je připraven tolik riskovat a použít strategii A. Jedním z cílů při výběru strategií s náhodnými výsledky je koncept náhodné dominance. Strategie A dominuje strategii B, pokud pro všechny možné výsledky x je pravděpodobnost, že výsledek převyšuje x je větší pro strategii A než pro strategii B. K nadefinování této podmínky matematicky, potřebujeme distribuční funkci strategie A jako $F_A(x) = p(\text{výsledek} < x)$ a také $F_B(x)$. Potom strategie A dominuje strategii X, jestliže $F_A(x) < F_B(x)$. To implikuje, že graf $F_A(x)$ vzhledek k x leží napravo od grafu $F_B(x)$, což je zobrazeno na obrázku 2.7a.

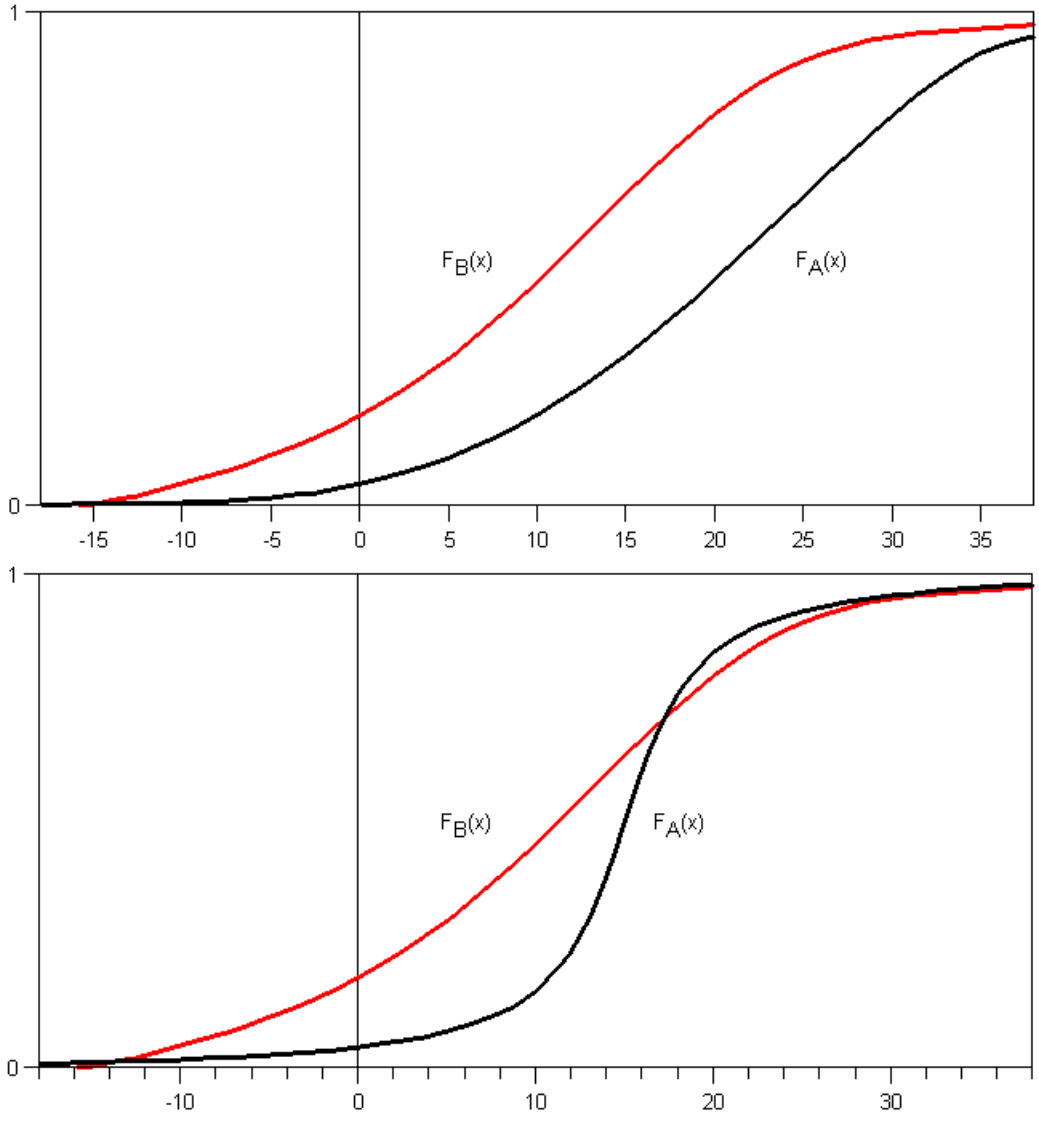
Teorie náhodné dominance byla rozšířena k pokrytí případu, kdy ani A nedominuje B ani B nedominuje A, ve výše uvedeném smyslu. Například druhý řád nahodné dominance se projeví pro A pokud

$$\int^r F_B(x)dx > \int^r F_A(x)dx \quad : \forall r \in \Re,$$

To implikuje, že pro všechny výsledky do r plati $F_A(x) < F_B(x)$ (viz obrázek 2.7b). Přestože bychom očekávali, že si každý vybere strategii A dominující strategii B, ne každý se tak rozhodne, pokud uvidí druhý řád dominance. Lidé, kteří si vybírají strategii podle druhého řádu dominance se nazývají „risk averse“, protože to signalizuje bezpečnější sázku.

Poznámka

ICT umožňují experimentování s vyvíjeným matematickým modelem, tj. jak sledování účinků změn jednotlivých parametrů v modelu na výsledné chování systému pomocí analýzy citlivosti, tak i zjišťovat vliv různých rozhodnutí a zásahů řešitele na fungování systému. Vlastnosti objektivní reality poznáváme pomocí měření chování systému pomocí fyzických experimentů s jejích adekvátním zobrazením pomocí modelu. Toto experimentování má pak povahu procesu učení.



Obrázek 2.7: Porovnání strategií a jejich dominance

Kapitola 3

Modelování měkkých systémů

Jedním z důležitějších hledisek v matematickém modelování je dělení systémů na tvrdé a měkké [18]. Toto dělení je v současné světové literatuře poměrně běžné, a proto jej citujeme i my, i když v sobě skrývá určitou logickou nedůslednost - nerozlišuje reálný systém od jeho systémového popisu.

Systémový přístup při řešení ekologických a biologických problémů respektuje nejširší souvislosti mezi jevy v přírodě, a to se projevuje zcela novým, kvalitativním uvažováním a kvalitativní analýzou ekologického systému, kdy se soustřeďujeme i na oblasti zdánlivě s řešeným problémem nesouvisející: na souvislosti geografické, ekologické, krajinné, antropogenní, na tradice a zvyklosti. Systém, ve kterém se respektují tyto a podobné vlivy, se nazývá měkký systém. Modelování měkkých systémů ovšem vyžaduje zvláštní postupy a speciální podporu. Statistické a jiné kvantitativní metody se v měkkých systémech také užívají, ale podstatně se mění metodologie jejich použití. Měkký systém vymezujeme od tvrdého systému výčtem vlastností a zpravidla se odlišné vlastnosti staví vedle sebe. Jako ukázku lze uvést např. popis základních změn v přístupech k řešení problémů v průběhu historie, který zároveň vymezuje i některé podstatné vlastnosti měkkého systému, tabulka 1.4.

Měkká metodologie vyžaduje zcela jiný přístup ve všech fázích matematického modelování. Zdůrazňuje se především potřeba co nejúplnějšího poznání systému a jeho okolí a co nejvýstižnější popis problémů bez ohledu na možnosti kvantifikace nebo formálního popisu jevů. Někdy nelze systém formalizovat ryzími matematickými prostředky, a používají se proto různé jiné formy popisu. Systém a problém je např. možno popsat několika případovými studiemi, které nemusí zcela vystihovat všechny problémy, ale řeší problematiku z několika hledisek. Při řešení problému v měkkém systému nebývá výsledkem jednoznačné a konečné řešení. Řešení se může jevit jako doporučení a nemusí být jednoznačné. I výsledek, který pomáhá pochopit tendence změn v měkkém systému, může být akceptován jako úspěšný.

Pro řešení měkkých systémů je podle Checklanda (1976) třeba nahradit interdisciplinární koncepci transdisciplinární (transdisciplinary concepts). Transdisciplinární koncepce řešení problémů předpokládá překonání hranic i mezi značně vzdálenými vědními obory a sjednocení jejich poznatků.

Metodologie měkkých systémů je ve svém pojetí širší a metodologii tvrdých systémů obsahuje jako svoji součást. Rozmach měkkých metodologií je v posledních létech značný. Měkká metodologie je založena na tzv. metapřístupech systémové vědy a snaží se o do-

konalé vystižení vlastností systému, někdy i na úkor formální elegance zobrazení. Metodicky je to přístup nehomogenní, obecně nepřenosný, nedovoluje exaktní zjištění, zda bylo dosaženo optimálního exaktního řešení, uživatel se musí ve většině případů spokojit s dosti dobrým řešením. Měkká metodologie je humanistická v tom smyslu, že zahrnuje vliv lidského činitele uvnitř i v okolí systému v nejširších aspektech psychických, sociálních, sociologických a politických.

Jako ilustrace měkkých metodologií lze uvést tzv. Jenkinsův akční výzkum a Checklandovu metodologie měkkých systémů.

3.1 Jenkinsův akční výzkum

Akční výzkum podle Jenkinse [19] obsahuje čtyři fáze, připomínající Simonovo členění. Jsou to

1. identifikace
2. projekt
3. implementace
4. sledování provozu a případně nový projekt a nová implementace V každé fázi se respektují strategická pravidla:
   - identifikace se provádí kombinací tvrdých i měkkých postupů, využívá se expertů,
   - problém konfliktních situací se řeší metodami vah nebo jako vícekriteriální problém,
   - problém se strukturuje, postupuje se v iteracích,
   - konečný výstup se ověřuje v praxi pomocí experimentálního ověření, nebo pomocí expertů.

3.2 Checklandova metologie měkkých systémů

Checkland [18] vycházel ve své metodologii z Jenkinsových zkušeností. Na rozdíl od Jenkinse předpokládá, že jeden a týž problém může být řešen a hodnocen z různých pohledů různě a při řešení je tedy třeba současně postupovat nejméně ve dvou úrovních. Tyto dvě úrovně jsou:

- vhodný model reality s aktivním zapojením a angažovaností lidí (tj. nemusí se jednat o formální matematický model),
- abstraktní model vyšší úrovně. Po obou úrovních se postupuje souběžně ve vzájemné interakci.

Postup je rozpracován do 7 fází:

1. popis problémové situace,

2. strukturování a identifikace problému,
3. vymezení subsystémů, jejichž analýza povede k řešení problému - realizace CATWOE,
4. tvorba projektu, modelu,
5. tvorba koncepce, konceptuálního modelu,
6. výběr řešení a implementace,
7. sledování zpětné vazby a opravy řešení.

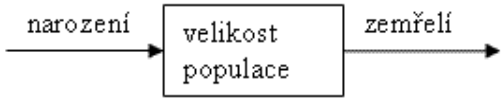
Metodologie tvrdých systémů vychází z matematických principů a matematických přístupů k analýze systému. V případě měkkých systémů se naopak hledají prostředky jak problém dokonale popsat a systém zobrazit, třeba i nematematickými prostředky a za vágních předpokladů. Skupiny takových metod a postupů se nazývají metametodologie. Prvním a nejdůležitějším úkolem metametodologie je popis postupu, který bychom mohli nazvat jako vytvoření metodologie metodologií. Od takového postupu se potom odvozují další metakonstrukce v systémové vědě jako jsou metametodologické dotazování, modelování, apod. Některé postupy metametodologie lze popsat matematicky, některé je nutno vyjádřit empiricky. Mezi empirické metody patří např. expertní analýza, vyjádření skutečnosti pomocí subjektivní pravděpodobnosti, použití fuzzy jazykových operátorů, simulace.

V souvislosti s rozvojem metametodologií (metamodelování, metasystémů) je vhodné uvést zajímavou klasifikaci věd a vědeckých přístupů k řešení problémů, kterou uvádí Rosen (1986) a která vymezuje postavení měkkého systému v současné vědě. Současnou vědu a její přístupy charakterizuje jako dvourozměrnou vědu (Two-Dimensional Science), což je věda v informačním prostředí. Průmyslová společnost produkuje jednorozměrnou vědu (One-Dimensional Science), která je charakteristická experimentováním. Rozvoj umělé inteligence přidá v příštím století vědě zřejmě další dimenzi.

Kapitola 4

Jednoduché modely

Mnoho procesů lze považovat za rozdělené na časové úseky, kdy zvažujeme přírůstek a úbytek za jednotku času. Typickým příkladem může být společenství lidí, ve kterém je přírustek reprezentován rozením nových jedinců a úbytek přirozeným vymíráním a časovou jednotkou může být měsíc nebo rok. Tento proces ilustruje obrázek 4.1.



Obrázek 4.1: Vstupně-výstupní diagram velikosti populace

Podobný princip můžeme sledovat i u těchto dalších modelů:

- proces radioaktivního rozpadu,
- znečištění prostředí, atmosféry,
- vstřebávání látek (léčiv) v těle, krvi.

Tato kapitola obsahuje několik modelů, jejichž společným rysem je možnost zapsání diferenciálními rovnicemi, které lze vyřešit elementárními metodami. Podrobné analytické řešení některých z těchto modelů lze nalézt v učebnici *Spojité modely v biologii* [9]. Mnoho dalších příkladů lze nalézt v následující literatuře: [1, 17, 8, 20, 21, 4, 22] a mnoha dalších.

4.1 Růstové modely

V tomto jako i dalších oddílech zabývajících růstem populací budeme značit[9, 1, 17] $x(t)$ velikost populace v čase t. Velikostí rozumíme nějakou jednotku relevantní k dané populaci (počet hlodavců, velikost porostu v m^2). Populací může být například společenstvo živých organismů, tráva rostoucí na louce, souhrn atomů radioaktivní látky, kapitálové zásoby podniku a podobně. Nechť $b(t, x)$ je přidání jedinců k populaci za jednotku času v čase t a při velikosti populace x (birth rate). Tato hodnota tedy určuje rychlost růstu dané

populace. Dále nechť $d(t,x)$ určuje rychlost vymírání dané populace v čase t v závislosti na velikosti populace x (death rate).

V dalším výkladu budeme předpokládat, že $b(t,x)$ i $d(t,x)$ jsou nezáporné spojité funkce a velikost populace je diferencovatelná funkce vzhledem k času.

Změnu velikosti populace za čas δt, které je velmi malé, lze zapsat:

$$\frac{x(t+\delta t)-x(t)}{\delta t} = b(t,x)x(t)-d(t,x)x(t)$$

Limitním přechodem $\delta t \to 0$ obdržíme diferenciální rovnici:

$$\frac{dx(t)}{dt} = g(t,x)x \tag{4.1}$$

kde $g(t,x) = b(t,x)-d(t,x)$ (growth rate). Funkce $g(t,x)$ je tzv. *specifická míra růstu*, která nemusí záviset pouze na čase a velikosti populace, ale i na řadě dalších faktorů. Rovnice 4.1 není matematickým modelem. Matematickým modelem se stává až po stanovení předpokladů kladených na specifickou míru růstu $g(t,x)$.

4.2 Model radioaktivního rozpadu

Proces určování stáří určitých artefaktů v našem okolí je důležitý k porozumění naší historie. Historici, geologové, paleontologové, archeologové, ale i další vědci využivají tyto informace ve svých oborech pro formulování různých teorií a ověřování hypotéz.

Existence nestabilních prvků, které se v průběhu času mění, je čtenáři jistě známa. Radioaktivita neboli radioaktivní rozpad je samovolná přeměna jader nestabilních nuklidů na jiná jádra, při níž vzniká ionizující záření. Změní-li se počet protonů v jádře, dojde ke změně prvku. Radioaktivitu objevil v roce 1896 Henri Becquerel u solí uranu. K objasnění podstaty radioaktivity zásadním způsobem přispěli francouzští fyzikové Pierre a Marie Curieovi.

Nejprve musíme předpokládat větší počet nuklidů, abychom se vyhnuli možné náhodnosti jevu, která by se mohla vyskytnout při malém počtu nuklidů. Dále musíme předpokládat jev probíhající kontinuálně v čase a nezvyšování hmoty zkoumaného materiálu. Použijeme-li diagram z obrázku 4.1, můžeme na základě předchozích předpokladů uvažovat, že zde není žádný vstup. Model nám tedy degraduje na jednodušší případ:

radioaktivní materiál v čase t+δt = radioaktivní materiál v čase t - množství rozpadlého materiálu za čas δt

Nechť populací v tomto modelu jsou radioaktivních atomy v nějakém izotopu chemického prvku. Velikostí této populace je aktuální počet těchto atomů v čase t.

V rámci zažitých konvencí budeme značit počet atomů N. Dále λ značí rozpadovou konstantu prvku (tato konstanta je nezáporná). Pro výše zavedené (rovnice 4.1) značení tedy máme $b(t,x) = 0$, $d(t,x) = \lambda$, $x(t) = N(t)$.

Dostáváme tedy:

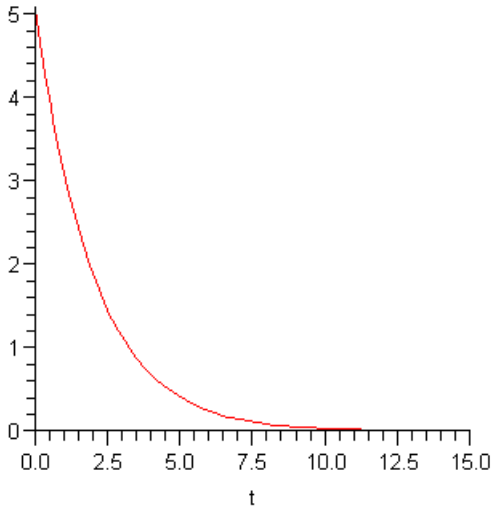
$$N(t+\delta t) = N(t) - \lambda N(t)\delta t.$$

Tato rovnice má známé řešení: $N(t) = N_0 e^{-\lambda t}$.

Řešení v Maple lze zapsat následujícím způsobem:

```
> restart:
> de := diff(N(t), t) = lambda*N(t):
> soln := dsolve(de, N(t0) = n0, N(t)):
> lambda := -0.5: n0 := 5: t0 := 0:
> plot(N(t), rhs(soln), t = 0 .. 15);
```

Výsledný graf si lze prohlédnout na obrázku 4.2.



Obrázek 4.2: Graf radioaktivního rozpadu

4.2.1 Poločas rozpadu

Poločas rozpadu τ radioaktivního nuklidu může být použit k výpočtu λ. Poločas rozpadu určuje dobu potřebnou k rozpadu poloviny radioaktivního materiálu a je všeobecněji známější než rozpadová konstanta λ.

Poločasy rozpadu mnoha látek již byly určeny. Například poločas rozpadu uhlíku (^{14}C) je 5568 let a uranu (^{238}U) 4,5 miliardy let.

Výpočet poločasu rozpadu

Použijeme-li řešení modelu a dále zapíšeme předpoklad o poločasu rozpadu:

```
> N(t+tau)=N(t)/2;
```

$$N(t+\tau) = \tfrac{1}{2}N(t)$$

Dalším výpočtem získáme:

```
> %/N(t);
> N:=x→N[0]*exp(-lambda*(x-t[0]));
> simplify(%%);
```

$$e^{-\lambda\,\tau} = 1/2$$

zlogaritmováním obou stran rovnice a vyřešením dostaneme závislost λ na τ:

```
> -lambda*tau=ln(1/2):
> %*(-1):
> lambda=solve(%, lambda);
```

$$\lambda = \frac{\ln(2)}{\tau}$$

Jak je vidět, poločas rozpadu τ a rozpadová konstanta λ nejsou závislé na počáteční velikosti radioaktivního materiálu N_0 ani t_0.

4.2.2 Příklad určení stáří maleb

Model radioaktivního rozpadu se stal v polovině minulého století jediným důkazem [1] pro usvědčení známého holandského malíře H. A. van Meegerena z padělání obrazů holandského mistra ze 17. století Johanna Vermeera. Van Meegeren studoval a znal velmi dobře všechny postupy používané při ověřování pravosti obrazu, a proto používal plátna ze starých bezcenných obrazů a také speciální druh barev, který smíchával dohromady s chemickými látkami tak, aby poté, co se obraz zahřál v troubě, ztvrdly a vytvořily tak dokonalou imitaci starého obrazu. Jediným způsobem, jak odhalit jeho padělky, bylo určení samotného stáří obrazu. Nekopíroval však známé obrazy, ale místo toho znovu kreslil Vermeerovy slavné obrazy, které v průběhu staletí zmizely. Namaloval například i Emauzské učedníky (obr 4.3), kteří byli několik let umístěni v Rotterdamu v Boymanově muzeu jako nejvýznamnější exponát muzea a jeden z nejvýznamnějších holandských obrazů. Dodnes existují lidé, kteří odmítají uznat, že jde o padělek.

Obrázek 4.3: Emauzští učedníci (padělek H. A. van Meegerena)

Pokud t_0 udává čas, kdy daná látka vznikla nebo byla vyrobena, potom stávající věk látky je $\frac{ln(\frac{N_0}{N})}{\lambda}$. Přeměnová konstanta λ je v mnoha případech známá nebo ji lze lehce vypočítat (stejně jako hodnotu N). Takže pokud známe také N_0, můžeme již snadno vypočítat stáří dané látky. Protože N_0 většinou neznáme, můžeme pro něj v mnoha případech

určit alespoň určitý rozsah, jako právě v případě van Meegerenových falšovaných obrazů. Radioaktivní přeměna je umožněna tím, že daná látka obsahuje určité množství uranu. Ten se přeměňuje na jinou radioaktivní látku a ta zase na jinou, až do doby, kdy je řetězec zakončen přeměnou na neradioaktivní olovo (^{206}Pb). Barvy používané malíři již přes 2000 let se skládají z bílého olova, které obsahuje malé množství radioaktivního radia (^{226}Ra) s poločasem rozpadu 1600 let a olova (^{210}Pb) s poločasem rozpadu 22 let. Nechť $N(t)$ a N_0 vyjadřují množství olova (^{210}Pb) na gram bílého olova v časech t a t_0. Dále nechť $r(t)$ popisuje počet radioaktivních rozpadů radia (^{226}Ra) za minutu na gram bílého olova v čase t. Je-li λ přeměnová konstanta pro olovo (^{210}Pb), potom

```
> diff(N(t),t)=-lambda*N(t)+r(t);
```

$$\frac{d}{dt}N\left(t\right)=-\lambda\,N\left(t\right)+r\left(t\right)$$

Protože nás zajímá období nejvýše 300 let, lze díky vysokému poločasu rozpadu považovat množství radia za neměnné a $r(t)$ pak za konstantu r. Vynásobením obou stran integračním faktorem $e^{\lambda t}$ získáme

```
> diff(N(t),t)*exp(lambda*t)=r*exp(lambda*t);
```

$$\left(\frac{d}{dt}N\left(t\right)\right)e^{\lambda\,t}=re^{\lambda\,t}$$

```
> exp(lambda*t)*N(t)-exp(lambda*t[0])*N[0] = r/lambda*(exp(lambda*t)-
exp(lambda*t[0]));
```

$$e^{\lambda\,t}N\left(t\right)-e^{\lambda\,t_0}N_0=\frac{r\left(e^{\lambda\,t}-e^{\lambda\,t_0}\right)}{\lambda}$$

```
> N(t) = r*(1-exp(-lambda*(t-t[0])))/lambda+N[0]*exp(-lambda*(t-t[0]));
```

$$N\left(t\right)=\frac{r\left(1-e^{-\lambda\,\left(t-t_0\right)}\right)}{\lambda}+N_0e^{-\lambda\,\left(t-t_0\right)}$$

Hodnoty $N(t)$ a r mohou být snadno změřeny. Jelikož zjistit hodnotu N_0 je složité musíme upravit předchozí rovnici do následujícího tvaru:

```
> lambda*N[0] = lambda*N(t)*exp(lambda*(t-t0))-r*(exp(lambda*(t-t0))-
1);
```

$$\lambda\,N_0=\lambda\,N\left(t\right)e^{\lambda\,\left(t-t0\right)}-r\left(e^{\lambda\,\left(t-t0\right)}-1\right)$$

kde λN_0 je počet rozpadů pokud bylo bílé olovo vytěženo z rudy v čase t_0. Za očekávaných okolností se tato hodnota pohybuje okolo 30.000 rozpadů (za minutu na gram). Jelikož nepotřebujeme znát přesné stáří, ale chceme pouze odhadnout, zda je obraz starý zhruba 300 let nebo se jedná o novodobý padělek, můžeme pro předchozí rovnici určit $(t-t_0)$. Vycházíme přitom z toho, že pokud se jedná o starší kresbu, bude množství radioaktivity z olova téměř stejné jako z radia. Naopak, pokud se jedná o padělek starý přibližně 20-50 let, potom radioaktivita z olova bude značně větší než radioaktivita z radia. Předpokládejme tedy, že chceme určit, zda se skutečně jedná o kresbu starou 300 let. Nechť tedy $(t-t_0)=300$. Dosazením do rovnice poslední získáme:

```
> restart;
> xx:=lambda*N[0] = lambda*N(t)*exp(300*lambda)-r*(exp(300*lambda)-
1);
```

$$\lambda N_0 = \lambda N(t) e^{300\lambda} - r\left(e^{300\lambda} - 1\right)$$

Pokud se jedná o padělek, bude hodnota N_0 extrémně vysoká. Protože poločas rozpadu olova (^{210}Pb) je velmi podobný poločasu rozpadu polonia (^{210}Po), se kterým se lépe počítá, nahradíme jej v našich výpočtech poloniem. Přeměnová konstanta se vypočítá snadno podle rovnice $\lambda = \frac{ln(2)}{22}$. Proto když víme, že $r = 0.8$ a $\lambda N(t) = 8.5$, tak můžeme snadno spočítat N_0:

```
> xx/lambda;
> r:=0.8: lambda:=ln(2)/22: N:=t→8.5/lambda:
> solve(xx, N[0]*lambda);
```

$$N_0 = \frac{1387724.800\, 2^{\frac{7}{11}} + 17.6}{\ln(2)}$$

$$98050.26120$$

Což je nepřiměřeně vysoká hodnota svědčící o padělku.

4.3 Model růstu populace živých organismů

4.3.1 Malthusův model

Tento model, vytvořený angličanem Thomasem Malthusem (1766-1834), nepředpokládá závislost specifické míry růstu na velikosti populace. Tento model je dostatečným v kratším časovém úseku a při malé populaci, kdy dává velmi dobrou shodu se statistickými daty. Pokud měříme t v diskrétních bodech 0, 1, 2, ... dostáváme geometrický růst populace, který však postupem času přestává souhlasit, protože populace roste neomezeně a to není reálné. Při větší populaci tedy tento model nevyhovuje.

Rovnice modelu je následující:

$$\frac{dx(t)}{dt} = ax(t)$$

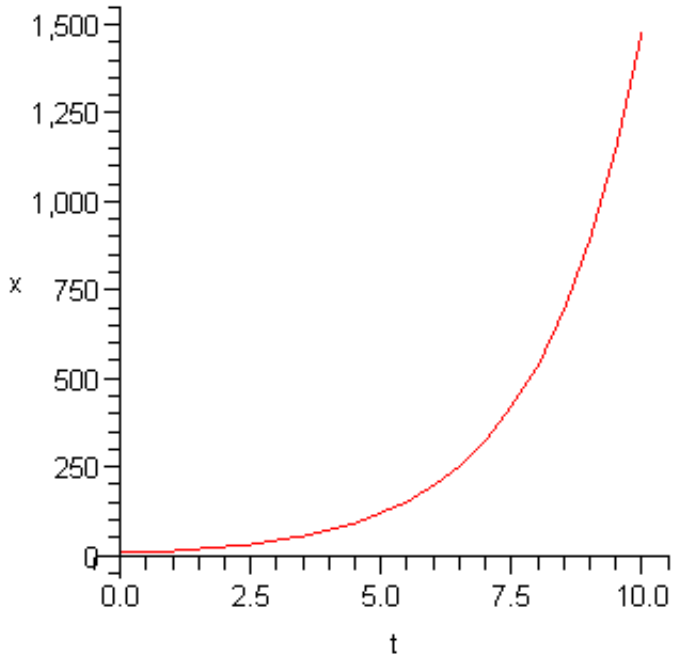
Řešení v Maple může být následující:

```
> malthus := proc(a, steps)
  local de1:
  de1 := diff(x(t), t) = a * x(t):
  DEplot(de1, x, t=0..steps, [[x(0)=10]], scene=[t, x], linecolor=red,
         linestyle=SOLID, thickness=1, stepsize=1, arrows=none):
end proc:
```

Růst populace je znázorněn na obrázku 4.4.

4.3.2 Logistická (Verhulstova) rovnice

Jelikož předchozí model nevyhovoval, byla ho potřeba upravit. Vzhledem k tomu, že jednotliví jedinci v populaci mezi sebou soupeří například o stravu dochází k *vnitrodruhové konkurenci*, která zvyšuje úmrtnost nebo snižuje porodnost. Vnitrodruhová konkurence roste při větším počtu soupeřících jedinců. Je tedy vhodné předpokládat specifickou míru



Obrázek 4.4: Graf řešení Malthusova modelu pro a = 0.5, x(0) = 10

> malthus(0.5, 10);

růstu jako klesající funkci velikosti populace. Jednoduchou funkcí splňující tato kritéria je lineární funkce $g(x) = a–bx$, kde a je koeficient růstu a jedná se o specifickou míru růstu v ideálním prostředí bez omezení zdrojů, $b > 0$ je koeficient vnitrodruhové konkurence.

Obdržíme tedy rovnici podobnou rovnici 4.5.

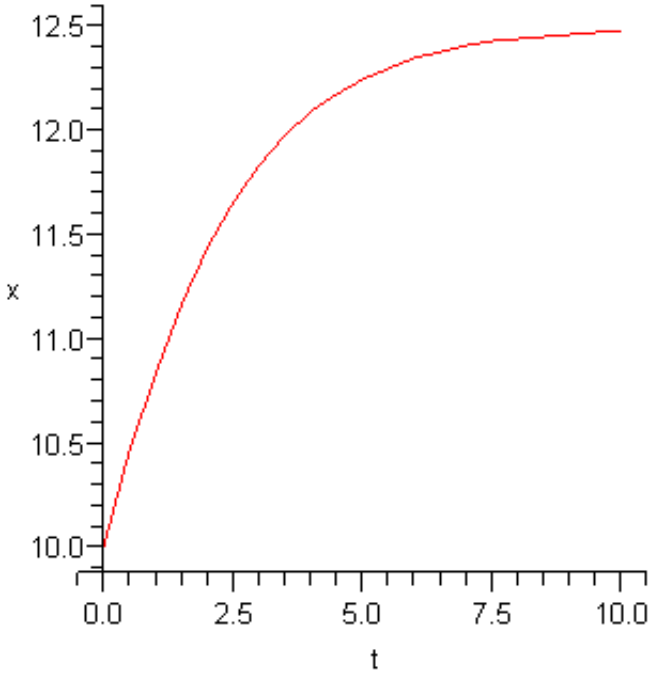
$$\frac{dx(t)}{dt} = (a - bx(t))x(t) \tag{4.2}$$

Rovnice 4.2 se nazývá logistická nebo Verhulstova podle belgického matematika Pierre François Verhulsta (1804 - 1849). Graf jejího řešení bývá nazýván logistická křivka. Tento název naznačuje podobnost s logaritmickou funkcí jak je vidět na obrázku 4.5.

Poznamenejme, že logistická rovnice se v literatuře objevuje i v následující podobě:

$$\frac{dx(t)}{dt} = rx(t)(1 - \frac{x(t)}{K}), \tag{4.3}$$

kde r je reprodukční růst (podobný dříve definovanému „growth rate“) a K je horní limita velikosti populace. Tuto rovnici použijeme v jednom z následujících odstavců (model sklizně, odstavec 4.5)



Obrázek 4.5: Graf řešení logistické rovnice pro a = 0.5, b = 0.04, x(0) = 10

4.3.3 Model růstu biomasy rybí populace

Nechť $x(t)$ značí hmotnost biomasy rybí populace v čase t. Předpokládejme, že se ryby živí planktonem. Nechť S značí hmotnost dostupného planktonu, který zbývá v čase t. Čím více je dostupného planktonu, tím větší je specifická míra růstu rostliny. Nejjednodušší předpoklad je $g(t,x) = kS(t)$, kde $k > 0$ je konstanta. Tato možnost samozřejmě není jediná – nabízí se nespočetné množství kombinací. Dosazením do rovnice 4.1 dostáváme:

$$\frac{dx(t)}{dt} = kS(t)x(t) \tag{4.4}$$

Pokud budeme předpokládat, že jednotka hmotnosti biomasy ryb se vytvoří spotřebováním m jednotek planktonu, dostáváme:

$$\frac{dS(t)}{dt} = -m\frac{dx(t)}{dt}$$

a odtud

$$S(t) = S(0) - mx(t)$$

Dosazením do rovnice 4.4 získáme:

$$\frac{dx(t)}{dt} = k(S(0) – mx(t))x(t) \tag{4.5}$$

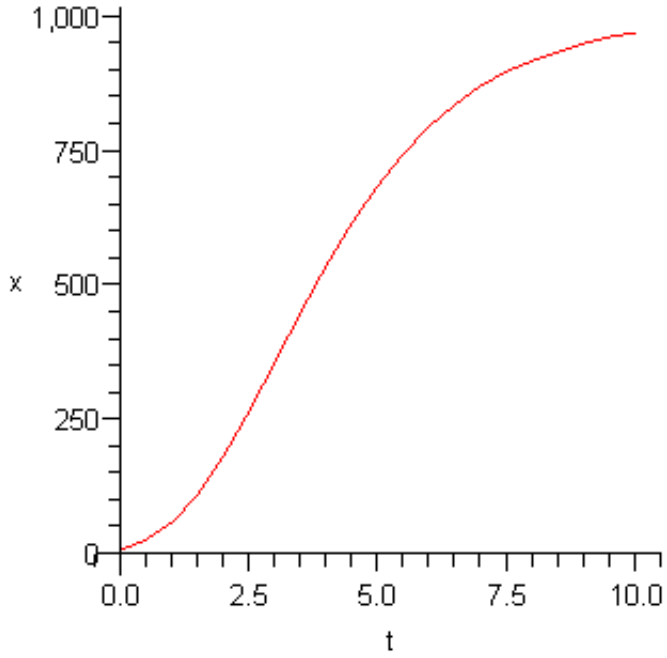
Obdrželi jsme tak model, kde specifická míra růstu závisí na velikosti populace x.

4.3.4 Gompertzova křivka

Řešením další možné funkce pro určení vnitrodruhové konkurence je tzv. Gompertzova křivka (britský matematik Benjamin Gompertz (1779 - 1865)). Specifická míra růstu je zadána rovnicí:

$$g(x) = -a \ln \frac{x}{K}$$

kde a, K jsou kladné parametry. Pokud se x limitně blíží k nule zprava je hodnota této funkce nekonečno. Interpretace této vlastnosti může být následující: Pokud je populace natolik malá, že jí hrozí vyhynutí, začne se bránit zvýšeným množením. Tato vlastnost se využívá například při modelování růstu rakovinného nádoru. Vývoj populace při omezení Gompertzovou křivkou je zobrazen na obrázku 4.6.



Obrázek 4.6: Graf řešení Gompertzovy křivky pro a = 0.5, K = 1000, x(0) = 10

4.4 Model regulace glykémie inzulínem

Glykémie je koncentrace neboli hladina glukózy v krvi nebo v krevním séru. Lidské tělo je uzpůsobeno tak, aby glykémii udržovalo v poměrně stálém rozmezí cca 4-6 mmol/l (na lačno). Toto stálé rozmezí je dáno závislostí řady orgánů na glukóze, zejména mozku, jehož jediným zdrojem energie je právě glukóza. Na řízení hladiny glukózy v krvi se podílejí vlivy hormonální, autoregulační a nervové. Nejdůležitějším a nejpotřebnějším hormonem pro správnou hladinu glykémie je inzulín, který jako jediný glykémii snižuje. Glukóza se ukládá do jater odkud je postupně uvolňována do krve. Odtud se dále dostává do tkáně,

kde je spotřebována. Vylučování inzulínu ze slinivky břišní je stimulováno glykémií a vyloučený inzulín postupně z organismu zmizí.

Označme $G(t)$ glykémii v čase t a $I(t)$ koncentraci inzulínu v krvi v čase t.

A dále předpokládejme:

1. Rychlost uvolňování glukózy do krve je konstantní. Tuto rychlost označme α
2. Množství glukózy, která vstoupí do tkáně za jednotku času je přímo úměrná koncentraci inzulínu v krvi. Tuto konstantu úměrnosti označme β
3. Množství inzulínu vyloučeného ze slinivky do krve za časovou jednotku je přímo úměrné okamžité glykémii. Tuto konstantu označme γ
4. Rychlost rozkládání inzulínu je přímo úměrná jeho koncentraci. Tuto konstantu označme δ.

Na základě těchto předpokladů můžeme vytvořit následující matematický model obsahující dvě diferenciální rovnice.

$$\frac{dG(t)}{dt} = \alpha - \beta I(t)$$
$$\frac{dI(t)}{dt} = \gamma G(t) - \delta I(t)$$

4.5 Model sklizně

Sklízení populace konstantně nebo progresivně je velmi důležité pro mnohé obory. Příklady mohou být rybí farma, louka s trávou atd. Při modelování tohoto systému si můžeme pokládat otázky: Způsobí vysoká úroveň sklízení zničení populace? Může sklízení malého dílu populace způsobit životaschopnost oboru? Popišme si tento model nejprve slovně: *změna populace = noví jedinci - zemřelí jedinci - zemřelí jedinci nedostatkem prostoru - sklizení jedinci*

Předpokládejme dále, že velikost sklizně je konstantní. Použitím logistické rovnice pro předchozí popis dostaneme tuto diferenciální rovnici popisující modelovaný systém:

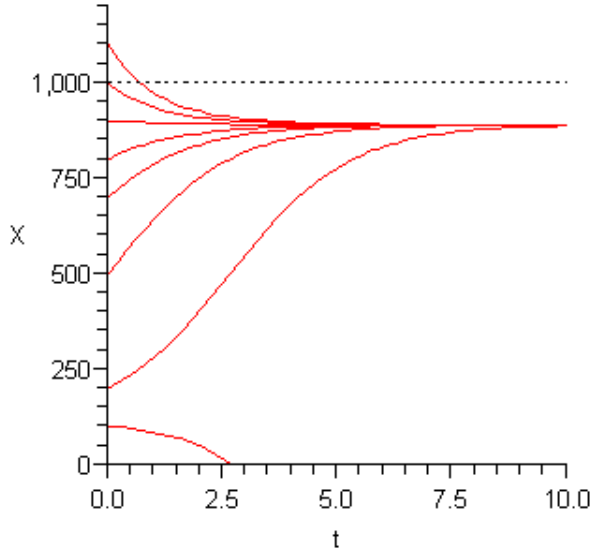
$$\frac{dx(t)}{dt} = rx(t)(1 - \frac{x(t)}{K}) - h, \tag{4.6}$$

kde h je konstantní míra sklízení (celkový počet sebraných jedinců nebo zemřelých kvůli sklízení za jednotku času) a je nezávislá na velikosti populace.

Příklad Ukažme si řešení tohoto modelu pro $K = 1000, h = 100, r = 1$

```
> rov := diff(X(t), t) = X(t) * (1 - X(t)/1000) - 100;
  hodnoty := [100, 200, 500, 700, 800, 900, 1000, 1100];
  for i from 1 to nops(hodnoty) do
    sol := dsolve([rov, X(0)=hodnoty[i]], numeric);
    graf[i] := plots[odeplot](sol, 0..10, color = red);
  end do:
  limita := plot(x->1000, 0..10, color=black, linestyle=DOT):
  plots[display]({limita, seq(graf[j],j=1..nops(hodnoty))}, view=
    [0..10, 0..1200]);
```

Výsledný graf je zobrazen na obrázku 4.7



Obrázek 4.7: Graf řešení modelu sklizně pro několik počátečních hodnot

4.6 Cvičení

1. Rybí farma Předpokládejme rybí farmu, kde jsou ryby loveny jednou týdně. Během každého výlovu je uloveno 2500 ks ryb. Míra růstu této populace je 0,6 ryby denně pro jednu rybu. Dále nechť míra úmrtnosti ryb je 0,1 ryby denně pro jednu rybu.

a) Zformulujte slovní popis rovnice růstu populace ryb. Dále vytvořte v Maple diferenciální rovnici pro počet ryb a graficky vyřešte.

b) Odhadněte počet uhynulých ryb po prvním týdnu, pokud je počáteční stav populace 300.000 kusů.

c) Jak musíme změnit velikost výlovu, abychom udržovali populaci na stejné hladině?

2. Myši v domě Uvažujme populaci 100 myší v domě, která má míru růstu 7 myší na jednu myš za měsíc a míru úmrtnosti 2 myši na jednu za měsíc. Dále je v domě nastraženo 15 pastiček na myši, které jsou každý týden plné. Popište tento systém modelem vyjádřeným diferenciální rovnicí.

3. Predikce modelem Uvažujme následující model:

$$\frac{dx(t)}{dt} = 0.3x(t) - 0.004x(t)^2$$

Předpokládejme počáteční velikost populace $x_0 = 200$ a časový krok jeden týden. Určete velikost populace po třech týdnech.

4. Model sklizně Uvažujme model sklizně z odstavce 4.5:

$$\frac{dx(t)}{dt} = rx(t)(1 - \frac{x(t)}{K}) - h.$$

a) Určete hladinu h_c pro parametry uvedené v odstavci 4.5, při které začne populace vymírat.

b) Zamyslete se, zda existuje počáteční hladina x_c, pro kterou i sklizeň $h < h_c$ vede k vymření populace. Ukažte, že tato počáteční hladina má velikost:

$$x_c = \frac{K}{2}(1 - \sqrt{1 - \frac{4h}{rK}})$$

5. Určení stáří kreseb v jeskyni Představte si, že jste objevili jeskyni ve které jsou kresby u kterých předpokládáte velmi historický původ. Tyto kresby obsahují zbytky živočisného uhlí. Dozvědeli jste se o možnosti určit stáří těchto kreseb pomocí obsahu radioaktivního uhlíku v tomto živočišném uhlí. Předpokládáte-li konstatní poměr radioaktivního uhlíku ve vzduchu v průběhu celé existence planety, lze uvažovat i tento poměr v živých organismech. Po smrti daného organismu již tělo dále nevstřebává radioaktivní uhlík, ale ten stávající podléhá radioaktivnímu rozpadu. Dále víte, že poločas rozpadu radioaktivního uhlíku (^{14}C) je zhruba 5568 ± 30 let. Uhlík obsažený v kresbách v jeskyni se rozpadá rychlostí 1,69 rozpadu za minutu na gram uhlíku. Dnešní stav radioaktivního uhlíku v ovzduší je 13,5 rozpadu za minutu na gram uhlíku.

a) Určete stáří těchto maleb. Řešení ověřte v systému Maple.

Pozn: Proces určování stáří touto metodou lze využít pro objekty staré zhruba 200 - 100.000 let. Na kratší časové úseky je vhodnější použít metodu uvedenou v odstavci 4.2.

Kapitola 5

Modely soužití dvou a více populací

V této kapitole se zaměříme na více populací, které se vzájemně ovlivňují. Pokud by tyto populace byly vzájemně nezávislé, můžeme je modelovat pomocí předchozích růstových modelů. V reálném prostředí však populace na sobě vzájemně závisí. Nejprve se podíváme na model dvou druhů, kdy se první stává potravou druhého a nazývaný model dravec-kořist (predator-prey). Nadefinujeme jednoduchý model označovaný jako Lotka-Volterrův po vědcích, kteří tento model poprvé publikovali. Dále potom tento model upravíme a zpřesníme.

Dalším modelem bude symbióza dvou populací, kdy si dva druhy navzájem pomáhají. Posledním modelem této kapitoly bude model konkurence dvou populací. Může se jednat o soupeření zvířat o omezenou potravu nebo například model války lidí.

Mnoho dalších modelů, jejich rozšíření případně jiné přístupy k řešení lze nalézt v následujících knihách: [9, 17, 1, 8, 20, 21, 10]

5.1 Model dravec-kořist

Uvažujme tedy dvě populace. Nechť první populace je kořistí a počet jejich jedinců v čase t budeme značit $X(t)$. Druhá populace bude dravcem a počet jedinců tohoto druhu v čase t označme $Y(t)$. Dále $\alpha_1 > 0$ je rychlost růstu velikosti populace kořisti a $\alpha_2 > 0$ je rychlost vymírání populace dravce.

5.1.1 Klasický Lotka-Volterrův model

Tento model byl vytvořen ve dvacátých letech minulého století nezávisle dvěma vědci – americkým matematikem Alfredem Jamesem Lotkou (1880-1949) a italským matematikem Vito Volterrem (1860-1940).

Oba vycházeli z následujících předpokladů:

1. Prostředí je idealizované – oba druhy existují odděleně od všech okolních vlivů a dravec se živí výhradně kořistí

2. Při neexistenci druhu dravce roste populace kořisti exponenciálně (konstanta α_1)

3. Při neexistenci druhu kořisti populace dravce vymírá (konstanta α_2)

4. Kořist vymírá působením dravce – je použita lineární funkce, kterou lze vyložit jako čím je větší populace kořisti, tím více jí je sežráno. Jelikož tento předpoklad není omezen, v praxi tato situace nenastává. Přirozené vymírání je zahrnuto v dynamice růstu populace. (konstanta β_1)

5. Dravec se množí na základě ulovené kořisti (konstanta β_2)

Matematický popis těchto předpokladů je následující:

$$\frac{dX(t)}{dt} = \alpha_1 X(t) - \beta_1 X(t)Y(t) \tag{5.1}$$

$$\frac{dY(t)}{dt} = -\alpha_2 Y(t) + \beta_2 X(t)Y(t) \tag{5.2}$$

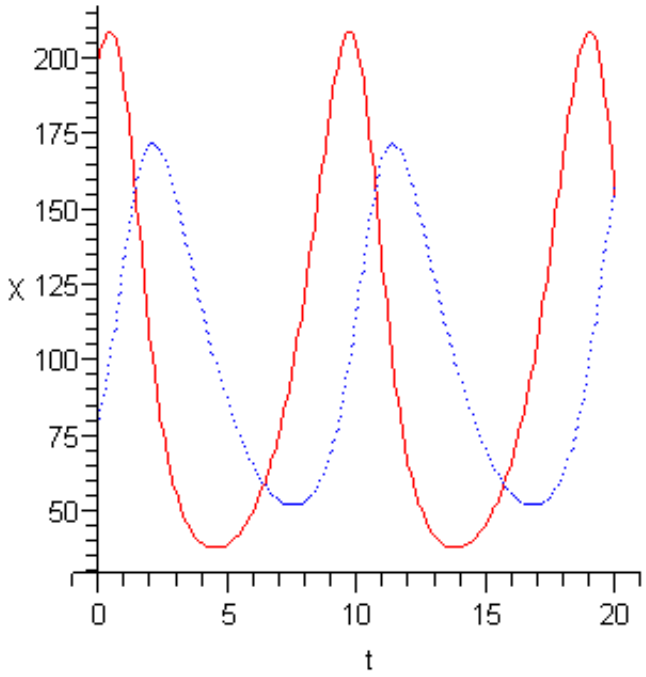
Řešení lze v Maple zapsat následujícím způsobem:

```
> lotka_volterra := proc(alpha1, alpha2, beta1, beta2, initPred,
                         initPrey, step)
  local eqPrey, eqPred, init, opts, pred, prey:

  eqPrey := diff(X(t), t) = alpha1*X(t) - beta1*X(t)*Y(t);
  eqPred := diff(Y(t), t) = -alpha2*Y(t) + beta2*X(t)*Y(t);

  init := [X(0) = initPrey, Y(0) = initPred];
  opts := stepsize=0.1, arrows=none, thickness=1:
  prey := DEplot([eqPrey, eqPred], [X, Y], t=0..step, [init],
              scene=[t, X], linecolor=red, linestyle=SOLID, opts):
  pred := DEplot([eqPrey, eqPred], [X, Y], t=0..step, [init],
              scene=[t, Y], linecolor=blue, linestyle=DOT, opts):
  display(prey, pred);
end proc:
```

Závislost množství kořisti [23] na množství predátora lze ve zkratce vyjádřit následujícím způsobem: Čím víc přibývá kořisti, tím víc s jistým zpožděním začne přibývat predátor. Větší množství predátora zvýší tlak na kořist a té tak začne ubývat. Posléze s klesajícím množstvím kořisti začne klesat i množství predátorů, kterým ubývá potrava. S ubývajícím množství predátorů se pokles kořisti zastaví a její množství začne opět stoupat. Klasické oscilace predátora a kořisti byly v přírodě popsány především v případech, kdy predátor má jen jednu převažující kořist: tedy především vlk + zajíc bělák a liška polární + lumík v polárních oblastech. Délka oscilace kolísá mezi 6 a 10 lety podle podmínek prostředí. Řešení pomocí Maple lze vidět na obrázku 5.1.

Analytické řešení

Z obrázku 5.1 je patrné, že velikosti obou populací v čase oscilují pro zvolené počáteční parametry. Zde si ukážeme proč k tomuto jevu dochází. Použijeme následující kód Maple pro vygenerování směrového pole:



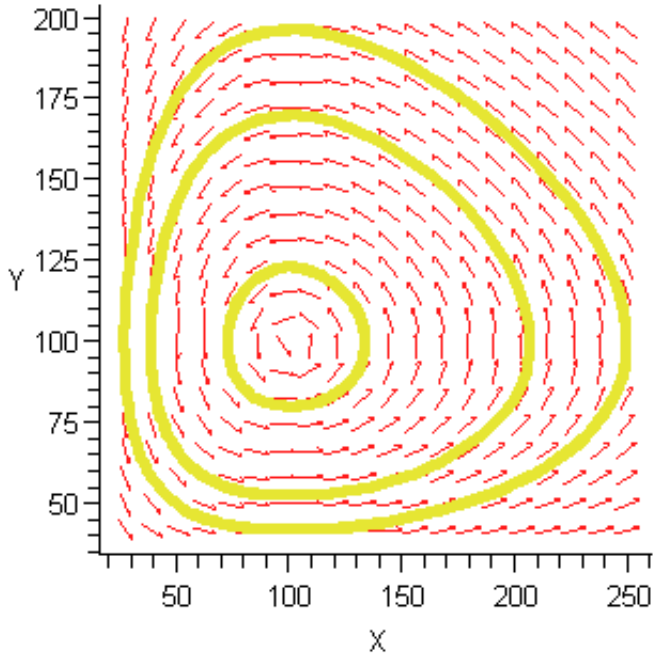
Obrázek 5.1: Řešení Lotka-Volterrova modelu

> lotka_volterra (1.0, 0.5, 0.01, 0.005, 200, 80, 20);

```
> alpha1 := 1.0:
  alpha2 := 0.5:
  beta1 := 0.01:
  beta2 := 0.005:
  eqPrey := diff(X(t), t) = alpha1*X(t) - beta1*X(t)*Y(t);
  eqPred := diff(Y(t), t) = -alpha2*Y(t) + beta2*X(t)*Y(t);

  init := [0, 100, 80], [0, 50, 50], [0, 100, 170]:;
  opts := stepsize=0.1:
  DEplot([eqPrey, eqPred], [X, Y], t=0..15, [init], scene=[X, Y],
              opts):
```

Výsledný graf je možno pozorovat na obrázku 5.2

Fakt, že křivky zobrazené na obrázku 5.2 jsou uzavřené, je velmi důležitý. Z počátečního bodu $[X_0, Y_0]$ (což jsou počáteční velikosti populací) můžeme postupovat ve směru zobrazených šipek a získáváme tak velikosti populací v průběhu času. Po dostatečně dlouhé době se opět dostaneme do počátečního stavu, což přesně odpovídá předchozím zjištěním z časově závislého grafu (obr. 5.1). Nyní se můžeme zaměřit na řešení obecného modelu a co o něm můžeme říci. Jak můžeme vidět pro různé počáteční hodnoty dostáváme různé křivky. Představme si křivku zredukovanou do jednoho bodu. V tomto bodě se tedy velikosti populací nemění a je řešením těchto diferenciálních rovnic. Toto řešení můžeme najít analyticky:



Obrázek 5.2: Závislost populací dravce a kořisti – směrové pole a trajektorie v okolí singulárního bodu

Nejprve pro rovnice 5.1 nastavíme $\frac{dX(t)}{dt} = 0$ a $\frac{dY(t)}{dt} = 0$ a obdržíme rovnice:

$$\alpha_1 X(t) - \beta_1 X(t) Y(t) = 0$$
$$-\alpha_2 Y(t) + \beta_2 X(t) Y(t) = 0$$

Abychom snadněji nalezli řešení převedeme je do upravené podoby:

$$X(\alpha_1 - \beta_1 Y) = 0, \quad Y(-\alpha_2 + \beta_2 X) = 0 \tag{5.3}$$

Z první rovnice je zřejmé, že řešením je $X = 0$ nebo $(\alpha_1 - \beta_1 Y) = 0$. Podívejme se tedy na obě možnosti.

$X = 0$ Dosazením do druhé rovnice okamžitě získáme $Y\alpha_2 = 0$ a tedy $Y = 0$. Dostali jsme tedy první řešení (X, Y) = (0, 0).

$(\alpha_1 - \beta_1 Y) = 0$ Úpravou dostaneme $Y = \frac{\alpha_1}{\beta_1}$. Dosazením do druhé rovnice obdržíme $-\alpha_2 + \beta_2 X = 0$, z čehož získáme řešení $X = \frac{\alpha_2}{\beta_2}$. Získali jsme tedy druhé řešení (X, Y) = $(\frac{\alpha_2}{\beta_2}, \frac{\alpha_1}{\beta_1})$.

Abychom se ujistili, že máme opravdu všechna řešení měli bychom stejný postup provést i pro druhou rovnici. V tomto případě však dostaneme opět stejná řešení a rovnice 5.3 mají tedy dvě řešení.

Podívejme se nyní na řešení v Maple

```
> restart;
  eqPrey := alpha1*X(t) - beta1*X(t)*Y(t);
  eqPred := -alpha2*Y(t) + beta2*X(t)*Y(t);
  solve({eqPrey, eqPred}, {X(t), Y(t)});
```

$$\{\mathrm{Y(t)} = 0,\ \mathrm{X(t)} = 0\}, \{\mathrm{Y(t)} = \tfrac{\alpha_1}{\beta_1},\ \mathrm{X(t)} = \tfrac{\alpha_2}{\beta_2}\}$$

5.1.2 Rozšířený Lotka-Volterrův model

Model uvedený v předchozím odstavci předpokládá jeden velmi nereálný předpoklad a to neomezený exponenciální růst populace kořisti v nepřítomnosti dravce. Tento nedostatek je zřejmý například z omezenosti prostoru, kde se kořist může vyskytovat nebo omezením její potravy. Je tedy nutné zavést nějaké omezení tohoto typu.

Označme tento limit K. Modifikované rovnice Lotka-Volterrova modelu mají následující tvar:

$$\frac{dX(t)}{dt} = \alpha_1 X(t)(1 - \frac{X(t)}{K}) - \beta_1 X(t)Y(t)$$
$$\frac{dY(t)}{dt} = -\alpha_2 Y(t) + \beta_2 X(t)Y(t)$$

Modifikace řešení v Maple je natolik snadná, že ji ponecháme čtenáři k procvičení. Obrázek 5.3 zobrazuje vývoj populací rozšířeného modelu.

5.1.3 Realističtější model Gauseho typu

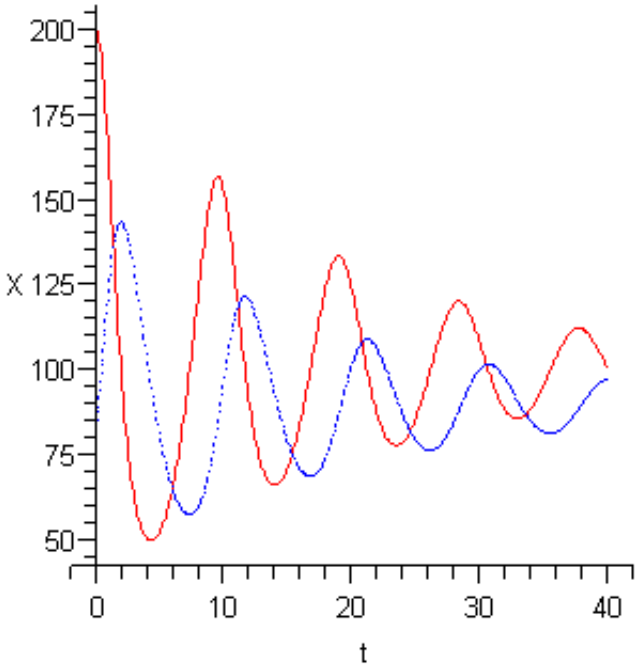
Oba předchozí modely mají stále ještě několik nevýhod: periodické řešení s libovolně velkou amplitudou, nižší počáteční stavy obou populací vedou k vysokým extrémům (vyzkoušejte si v předchozím modelu počáteční hodnoty 50, 10 nebo 10, 1), což opět není možné vzhledem k omezenosti zdrojů. Vzhledem k neustálým změnám a vnějším vlivům nedochází k pravidelnému opakovaní cyklů jak ukazují oba předchozí modely.

Pro tento model zavedeme následující předpoklady:

1. Vývoj populace kořisti izolované od dravce je úměrný její velikosti a specifická míra růstu závisí pouze na velikosti této populace. Zavedeme označení $g_1(X(t))$

2. Množství zabité kořisti závisí na její velikosti (nedostatek kořisti vede k jejímu delšímu vyhledání, přemnožená kořist se může bránit dravcům) i na velikosti populace dravce (mohou spolupracovat nebo vzájemně soupeřit). Zavedeme označení $d_1(X(t), Y(t))$

3. Kořist je jediným zdrojem potravy pro dravce. Pokud v prostředí není žádná kořist, dravec začne vymírat. Zavedeme označení g_2



Obrázek 5.3: Řešení rozšířeného Lotka-Volterrova modelu pro K = 1000

>

4. Dravci se mohou rozmnožovat pouze pokud mají k dispozici kořist, kterou ulovili. Zavedeme označení $d_2(d_1)$

5. Předpokládejme, že výše uvedené funkce jsou spojité.

Uvedené předpoklady vedou k následujícím rovnicím:

$$\begin{aligned} \frac{dX(t)}{dt} &= g_1(X(t))X(t) - d_1(X(t), Y(t))Y(t) \\ \frac{dY(t)}{dt} &= -g_2 Y(t) + d_2(d_1(X(t), Y(t)))Y(t) \end{aligned} \tag{5.4}$$

Pokud budeme dále předpokládat, že dravci navzájem nespolupracují ani nesoupeří, bude d_1 závislá pouze na prvním parametru. Pokud ještě budeme předpokládat, že množství ulovené kořisti se použije při růstu dravce (lze použít konstantu κ zavádějící vztah efektivní přeměny), lze tedy vyjádřit $d_2 = d_1\kappa$. Rovnice 5.4 potom mají následující tvar:

$$\begin{aligned} \frac{dX(t)}{dt} &= g_1(X(t))X(t) - d_1(X(t))Y(t) \\ \frac{dY(t)}{dt} &= -g_2 Y(t) + \kappa d_1(X(t))Y(t) \end{aligned} \tag{5.5}$$

Za g_1 můžeme zvolit některou z funkcí uvedenou v odstavci 4.2.2. Zůstává nám tedy k nadefinování pouze funkce d_1. Tato funkce se nazývá *trofická funkce* nebo *funkční odezva*

predátora. Jak je patrné z rovnic 5.5, předchozí dva modely vycházeli taktéž z těchto rovnic a použili exponenciální růst kořisti a jako trofickou funkci $d_1(X) = kX(t)$. Tato definice trofické funkce se však neosvědčila, jak bylo konzultováno výše, obzvláště svojí neohraničeností. Při velkém rozšíření kořisti dravec dosáhne nasycení ulovením určitého počtu jedinců kořisti a dále již neloví. Je proto potřeba zavést nějaké další požadavky na trofickou funkci:

- $d_1(0) = 0$ - dravec nemůže nic ulovit, jestliže populace kořisti neexistuje
- $\lim_{X\to\infty} d_1(X) = S$ - dravec uloví pouze S kusů kořisti a dále již neloví i při neomezené populaci kořisti
- d_1 je neklesající - při růstu populace kořisti dravec loví stejně nebo více

Existuje několik typů trofických funkcí [9]: [1]

Typ I Množství zabité kořisti je přímo úměrné množství dostupné kořisti až do hladiny nasycení. Při velkém množství kořisti trofická funkce nabývá hodnoty hladiny nasycení. Tuto funkci lze vyjádřit následovně:

$$d_1(X) = \begin{cases} aX & X < \frac{S}{a} \\ S & X \geq \frac{S}{a} \end{cases}$$

kde a je kladná konstanta. Tyto funkce jsou charakteristické pro živočichy lovící pomocí filtrování vody např. měkkýši.

Definice v Maple je následující:

```
> a := 10:
  d := x->piecewise(x<S/a, a*x, S);
  plot1 := plot(d, 0..20, thickness=2, color=yellow):
  display(plot1, limita);
```

Grafické řešení je zobrazeno na obrázku 5.4.

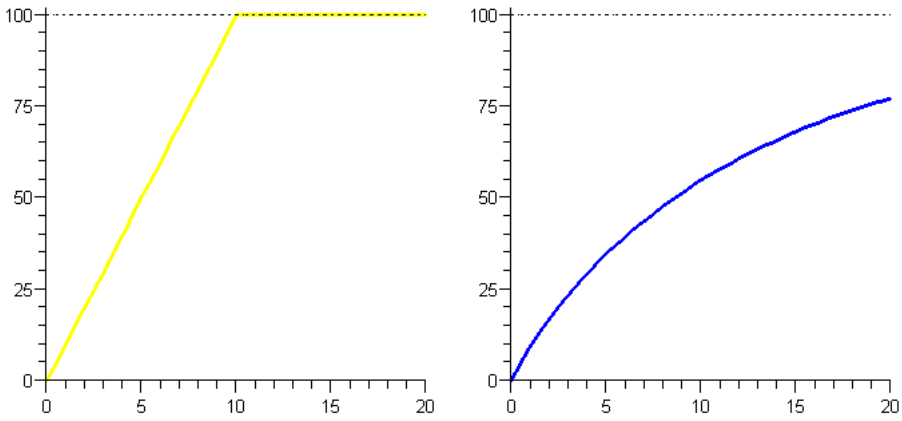
Typ II Jedná se o aproximaci funkcí typu I, kdy dochází k vyhlazení přechodu mezi nasyceným a skoro nasyceným stavem. Dravec tedy tento stav přísně nerozlišuje. Tyto funkce jsou charakteristické pro bezobratlé živočichy.

$$d_1(x) = S(1 - e^{-ax^k}), a \in R+, k \in (0, 1]$$

Grafické řešení je zobrazeno na obrázku 5.4 a definice v Maple vypadá následovně:

```
> a := 0.1:
  k := 0.9:
  d := x -> S*(1-exp(1)^(-a*x^k));
  plot2 := plot(d, 0..20, thickness=2, color=blue):
  display(plot2, limita);
```

[1] Pro všechny typy uvažujeme limit S=100 a při vykreslování grafu v Maple následující definici: limita := plot(x→S, 0..20, color=black, linestyle=DOT)



Obrázek 5.4: Trofická funkce typu I. (vlevo) a typu II. (vpravo)

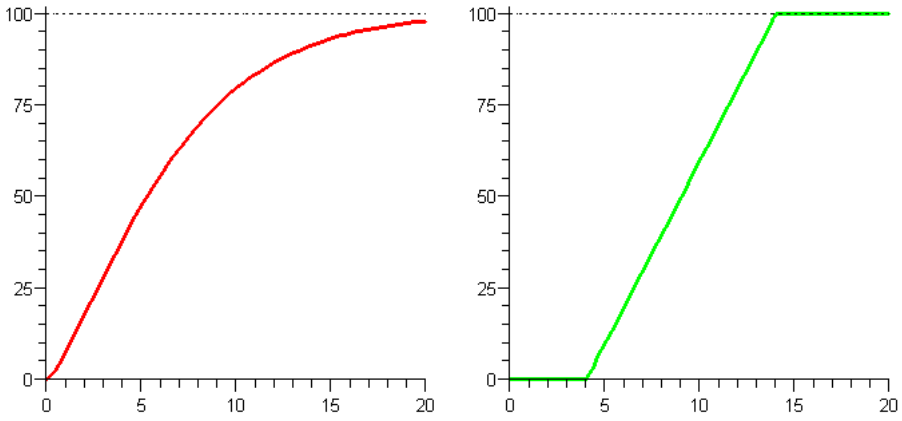
Typ III Jedná se o aproximaci funkcí typu IV, kdy dochází k vyhlazení obou přechodových zlomů.

$$d_1(x) = S(1 - e^{-ax^k}), k > 1$$

Definice v Maple je následující:

```
> a := 0.08:
  k := 1.3:
  d := x -> S*(1-exp(1)^(-a*x^k));
  plot3 := plot(d, 0..20, thickness=2);
  display(plot3, limita);
```

Grafické řešení je zobrazeno na obrázku 5.5



Obrázek 5.5: Trofická funkce typu III. (vlevo) a typu IV. (vpravo)

Typ IV Zde se projevuje problém pro dravce při nedostatku kořisti. Výskyt tohoto jevu může být způsoben možností kořisti skrýt se, ignorací kořisti dravcem pro její nedostatečný výskyt pokud existuje jiná dostupná strava atd. Funkce lze zapsat ve tvaru

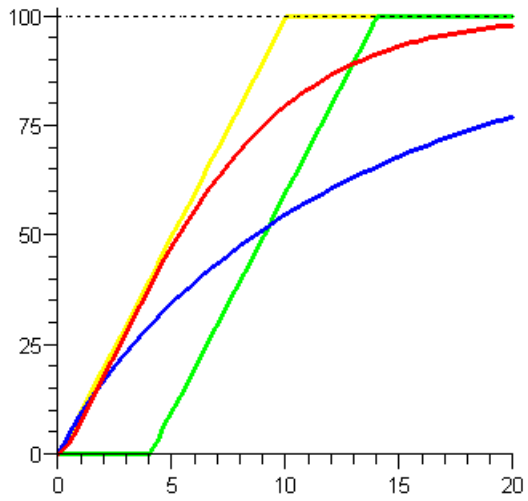
$$d_1(X) = \begin{cases} 0 & x < \frac{b}{a} \\ ax–b & \frac{b}{a} <= x <= \frac{S+b}{a} \\ S & x > \frac{S+b}{a} \end{cases}$$

kde a, b jsou kladné konstanty. Tyto funkce jsou charakteristické pro živočichy schopné složitějšího chování např. obratlovce.

Grafické řešení je zobrazeno na obrázku 5.5 Definice v Maple je následující:

```
> a := 10:
  b := 40: \#dolni limit
  d := x->piecewise(x < b/a, 0, b/a <= x and x<(S+b)/a, a*x-b, S);
  plot4 := plot(d, 0..20, thickness=2, color=green):
  display(plot4, limita);
```

Srovnání všech funkcí jsou uvedeny na obrázku 5.6.



Obrázek 5.6: Porovnání křivek trofických funkcí.

5.2 Model konkurence mezi dvěma populacemi

Dalším modelem koexistence dvou druhů je model soupeřících druhů, kde dvě a více populací soupeří o omezené zdroje - například prostor nebo potravu. Tento model je velmi podobný modelu dravec-kořist, avšak musíme o tomto systému uvažovat trochu jinak.

Předpoklady modelu

1. předpokládejme velikost populace dostatečně velkou, aby výpočet neovlivnil případný náhodný výkyv velikosti populace

2. předpokládejme, že model odpovídá systému dostatečně přesně

3. předpokládejme exponenciální růst populace při absenci ostatních soupeřících populací

Uvažujme vstupně-výstupní diagram velikosti populace (obr 4.1) pro dvě populace. Označme $X(t)$ a $Y(t)$ velikosti jednotlivých populací v čase t.

Rychlost růstu populace je dána pouze její aktuální velikostí a proto tento růst reprezentují kladné konstanty α_1 a α_2 pro populaci X resp. Y. Tyto konstanty opět obsahují přirozený úbytek populací, který se nevztahuje k soupeření mezi populacemi.

Protože oba druhy mezi sebou soupeří o stejný zdroj a tím se navzájem omezují, má velikost jedné populace vliv na vymírání druhé. Kladné konstanty β_1 a β_2, určující rychlost vymírání populace X resp. Y, budou tedy ovlivněny velikostmi obou populací v aktuálním okamžiku.

Zformulujme nyní rovnice popisující velikost jednotlivých populaci v čase t:

$$\frac{dX(t)}{dt} = \alpha_1 X(t) - \beta_1 X(t)Y(t) \tag{5.6}$$

$$\frac{dY(t)}{dt} = \alpha_2 Y(t) - \beta_2 X(t)Y(t) \tag{5.7}$$

Rovnice 5.10 bývají v literatuře občas označovány jako Gausovy rovnice.

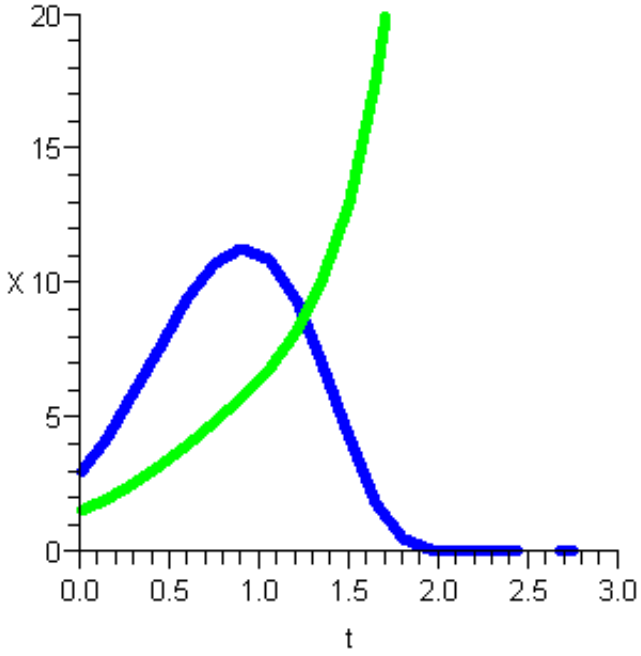
Přestože tyto rovnice nevypadají nikterak složitě, nemají analytické řešení, a proto zde uvádíme pouze numerické řešení závislé na čase v Maple:

```
> competing_species := proc(alpha1, alpha2, beta1, beta2, inX, inY,
    step)
  local eq1, eq2, init, opts, plot1, plot2;
  eq1 := diff(X(t), t) = alpha1*X(t) - beta1*X(t)*Y(t);
  eq2 := diff(Y(t), t) = alpha2*Y(t) - beta2*X(t)*Y(t);
  init := [X(0) = inX, Y(0) = inY];
  opts := method=rkf45, arrows=none:
  plot1 := DEplot([eq1, eq2], [X, Y], t=0..step, [init], scene=[t, X],
    linecolor=blue, opts):
  plot2 := DEplot([eq1, eq2], [X, Y], t=0..step, [init], scene=[t, Y],
    linecolor=green, opts):
  display(plot1, plot2, view=[0..3,0..20]);
end proc:
```

Výsledný graf je zobrazen na obrázku 5.7

Jak je z grafu 5.7 patrné jeden druh během času vyhyne. Změnou počáteční velikosti první populace ($X(0) = 5$) lze docílit přežití tohoto druhu. Výsledek o vyhynutí jednoho druhu, jenž odpovídá předpokladům, je znám jako *Gause's Principle of Competitive Exclusion*[24] (překládaný jako: *princip kompetitivního vyloučení*) . Tento princip byl ověřen na mnoha výzkumech v mikrobiologii a nedochází zde (narozdíl od modelu dravec-kořist) k periodické oscilaci velikosti populace v čase.

Stejně jako Lotka-Volterův model, je i tento model ovlivněn předpokladem exponenciálního růstu populace při obsenci soupeřů. Opět provedeme jeho rozšíření pomocí omezení maximální velikosti populací. I po této úpravě bude ale platit Gausův princip kompetitivního vyloučení a jedna z populací vyhyne.



Obrázek 5.7: Řešení modelu soupeřících druhů

```
> competing_species(3.5, 2.25, 0.6, 0.1, 3.0, 1.5, 3.0);
```

Předpokládejme tedy omezení růstu velikosti populace K_1 a K_2 pro populaci X resp. Y. Dále modifikujme rovnice 5.10

$$\frac{dX(t)}{dt} = \alpha_1 X(t)(1 - \frac{X(t)}{K_1}) - \beta_1 X(t)Y(t) \tag{5.8}$$

$$\frac{dY(t)}{dt} = -\alpha_2 Y(t)(1 - \frac{Y(t)}{K_2}) - \beta_2 X(t)Y(t) \tag{5.9}$$

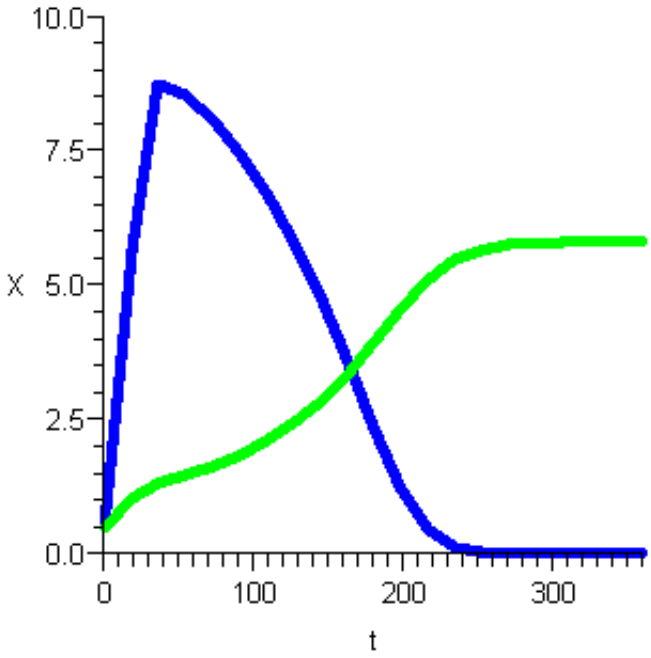
Úprava řešení v Maple je snadná a proto ji ponecháme čtenáři na procvičení. Výsledný graf je zobrazen na obrázku 5.8.

Příklad: Model bitvy Představme si výše zmíněné dva soupeřící druhy jako dvě armády [1] bojující proti sobě. V dobách před vynalezením střelných zbraní a zbraní hromadného ničení byla bitva soubojem muže proti muži ručními zbraněmi. Koeficient přežití zde však záleží i na mnoha dalších faktorech: zkušenost válečníků a jejich vůdců, trénink před bojem a další.

Naším cílem je vyvinout jednoduchý model, který bude počítat aktuální stav mužů v jednotlivých armádách, pokud známe počáteční stavy jednotlivých armád.

Nejprve si zadefinujeme několik dalších předpokladů:

- Předpokládejme dostatečně veliké armády, abychom vyloučili možné anomálie.



Obrázek 5.8: Řešení rozšířeného modelu soupeřících druhů (kapacity jsou 13 a 5.8, hodnoty vychází z Gausových pokusů)

> competing_species2(0.21827, 0.06069, 0.05289, 0.00459, 13, 5.8, 0.5, 0.5, 360);

- Předpokládejme, že armádám nepřichází žádné posily ani neztrácí své vojáky kvůli dezerci.
- V reálném boji se střílí přímo na nepřítele (cílená střelba) nebo do míst, kde by se nepřítel mohl vyskytovat (náhodná střelba). V některých bitvách může jeden z těchto druhů střelby dominovat. My budeme uvažovat cílenou střelbu pro obě armády.

Podobně jako v sekci o radioaktivním rozpadu (4.2) můžeme uvažovat vstupně-výstupní diagram velikosti populace z obrázku 4.1, který redukuje vstup a zůstává tedy pouze výstup, což jsou zemřelí vojáci. Při cílené střelbě předpokládáme střelbu „na cíl“. Počet zabitých vojáků tedy závisí na počtu střílejících neprátel a ne na počtu vojáků. To povede k zjištění, že v modelu budeme potřebovat konstantu určující úspěšnost střelby jednotlivých armád. Narozdíl při náhodné střelbě je počet zemřelých vojáků zavislý právě na počtu vojáků v dané lokalitě, kam se střílí a proto zde se uvažuje poměr mezi počtem střílících vojáků a ostřelovaných nepřátel.

Na základě předpokladů můžeme vytvořit následující diferenciální rovnice reprezentující náš systém:

$$\frac{dX(t)}{dt} = -\alpha_1 Y(t) \tag{5.10}$$

$$\frac{dY(t)}{dt} = -\alpha_2 X(t) \tag{5.11}$$

Pro řešení použijeme data z bitvy o Iwo Jima (1945) v Tichém oceánu [1] mezi armádou Japonska a USA. Američené měli přesnou statistiku o ztrátách po dni, proto z těchto dat byla vypočítány následující údaje:

- počáteční stavy: USA: $X(0) = 66454$, Japonsko: $Y(0) = 18274$.
- účinnost střelby: USA: $\alpha_1 = 0,0544$, Japonsko: $\alpha_2 = 0,0106$.

Vyřešme tento model numericky v Maple:

```
> battle:=proc(alpha1, alpha2, initX, initY, steps)
  local de1, de2, init, opts, usa, jap:
  de1 := diff(X(t), t) = -alpha1*Y(t):
  de2 := diff(Y(t), t) = -alpha2*X(t):
  init := [X(0) = initX, Y(0) = initY]:
  opts := stepsize=0.1, dirgrid=[10,10], arrows=none:
  usa := DEplot([de1, de2], [X, Y], t=0..steps, [inits], scene=[t, X],
     linecolor=blue, myopts):
  jap := DEplot([de1, de2], [X, Y], t=0..steps, [inits], scene=[t, Y],
     linecolor=red, myopts):
  display(usa, jap);
end proc:
```

A výsledný graf je na obrázku 5.9

5.3 Model symbiózy dvou populací

Následující model koexistence dvou druhů vychází z předchozího modelu soupeřících druhů. V tomto modelu se narozdíl od předchozího jednotlivé druhy doplňují místo toho aby spolu soupeřili. Tedy přítomnost jednoho druhu podporuje růst druhu druhého. Model je možné ještě doplnit o podmínky typy: „První populace nemůže žít bez druhé populace“, „Růst závislé populace se odvíjí pouze od velikosti populace hostitele“ atd.

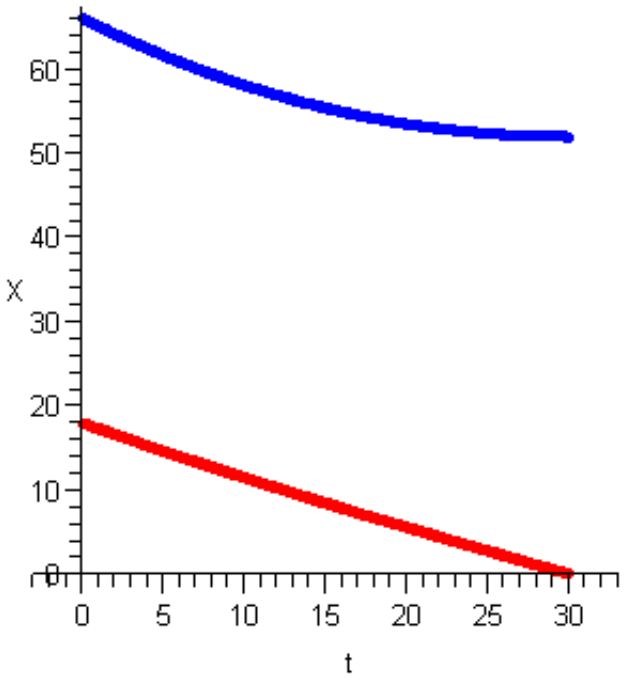
Předpoklady modelu jsou víceméně totožné s předpoklady předchozího modelu, pouze předpokládejme možný růst populace bez symbiotického druhu.

Uvažujme vstupně-výstupní diagram velikosti populace (obr 4.1) pro dvě populace. Označme $X(t)$ a $Y(t)$ velikosti jednotlivých populací v čase t.

Rychlost růstu populace je dána pouze její aktuální velikostí a proto tento růst reprezentují kladné konstanty α_1 a α_2 pro populaci X resp. Y. Tyto konstanty opět obsahují přirozený úbytek populací. Růst bez symbiotického druhu by měl být výrazně pomalejší než za přítomnosti symbiotického druhu.

Protože se druhy vzájemně podporují má velikost jedné populace vliv na další růst další populace. Kladné konstanty β_1 a β_2, určující rychlost růstu populace X resp. Y, budou tedy ovlivněny velikostmi obou populací v aktuálním okamžiku.

Zformulujme nyní rovnice popisující velikost jednotlivých populaci v čase t pro nejjednodušší model dvou závislých populací:



Obrázek 5.9: Řešení modelu bitvy

```
> battle(0.0544, 0.0106, 66, 18, 30);
```

$$\frac{dX(t)}{dt} = \alpha_1 X(t) + \beta_1 X(t)Y(t) \tag{5.12}$$

$$\frac{dY(t)}{dt} = \alpha_2 Y(t) + \beta_2 X(t)Y(t) \tag{5.13}$$

Stejně jako v předchozím případě nemají tyto rovnice analytické řešení, a proto zde uvádíme pouze numerické řešení závislé na čase v Maple:

```
> symbiosis := proc(alpha1, alpha2, beta1, beta2, inX, inY,
    step)
  local eq1, eq2, init, opts, plot1, plot2;
  eq1 := diff(X(t), t) = alpha1*X(t) + beta1*X(t)*Y(t);
  eq2 := diff(Y(t), t) = alpha2*Y(t) + beta2*X(t)*Y(t);
  init := [X(0) = inX, Y(0) = inY];
  opts := method=rkf45, arrows=none:
  plot1 := DEplot([eq1, eq2], [X, Y], t=0..step, [init], scene=[t, X],
    linecolor=blue, opts):
  plot2 := DEplot([eq1, eq2], [X, Y], t=0..step, [init], scene=[t, Y],
    linecolor=green, opts):
  display(plot1, plot2, view=[0..3,0..20]);
end proc:
```

Tento model je velmi idealizovaný a proto je potřeba ho doplnit o podmínky zmíněné na začátku sekce.

5.4 Cvičení

1. Lotka-Volterrův model Upravte rozšířený (o kapacitní omezení) Lotka-Volterrův model o další populaci a řešte v Maple.

a) Nechť je tato populace dalším dravcem a neovlivňuje se s druhou populací dravců.

b) Nechť je tato populace další kořistí a neovlivňuje se s druhou populací kořisti.

2. Model dravec-kořist s konkurencí mezi dvěma druhy dravce Upravte rozšířený (o kapacitní omezení) Lotka-Volterrův model o další populaci dravce, která bude bojovat o zdroj (kořist) s původní populací dravce a řešte v Maple.

3. Trofické funkce Zkuste pomocí nějaké vám známe metody vymyslet další možné rovnice jednotlivých trofických funkcí. Své řešení ověřte v Maple.

4. Model symbiózy dvou druhů Upravde model symbiózy dvou druhů následujícím způsobem:

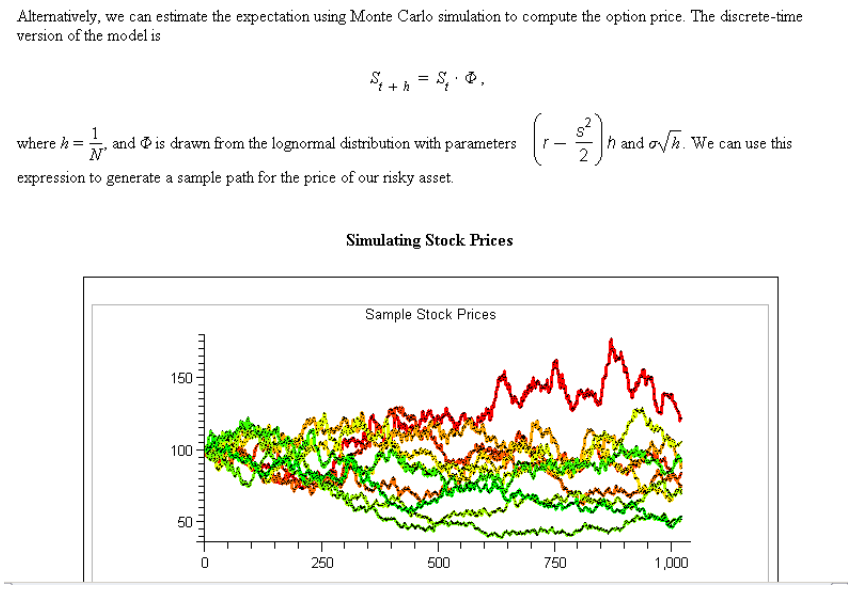
a) Množení prvního druhu je zcela závislé na existenci hostitelského druhu. Bez hostitele postupně vymírá. Při růstu však nesmí překročit omezení na velikost hostitelského druhu. Hostitelský druh není závislý na tomto druhu.

b) Stejné jako předchozí varianta avšak hostitelský druh roste rychleji v závislosti na počtu symbiotické populace.

Kapitola 6

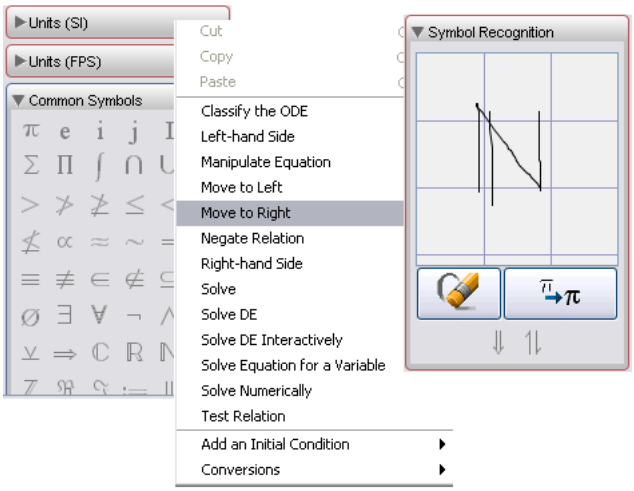
Maple

Maple se řadí do tzv. CAS systémů (Computer Algebra System) a je využíván nejen k výuce, ale své podstatné využití nachází též v oblasti vědy a výzkumu. Patří mezi technologické špičky ve své oblasti. Jeho poslední verze (Maple 10) se stala revoluční, díky svému přístupu k uživateli, kdy již není potřeba znát nazpamět spousty interních příkazů, samotný programovací jazyk nebo zdlouhavě vyhledávat v nápovědě. Systém uživatele navádí pomocí kontextových menu, palet nástrojů, šablon běžných problémů (task templates), nástroji pro rozpoznávání symbolů nebo interaktivních průvodců a vytváří tak interaktivní dokument. Náhled interaktivního dokumentu je zobrazen na obrázku 6.1. Nástroje pro ulehčení práce uživateli jsou ukázány na obrázku 6.2.



Obrázek 6.1: Interaktivní dokument Maple 10

Maple nyní umožňuje jednoduše vytvářet kompletní prezentace nejen matematických problémů, které mohou sloužit nejen k vyřešení problému, ale i jako dokumentace nebo k výukovým potřebám. Dalším rozšířením klasických worksheetů jsou Maplety. Jedná se o



Obrázek 6.2: Nástroje Maple 10 (vlevo panel symbolů, uprostřed kontextové menu, vpravo rozpoznávání symbolů)

grafické rozhraní podobné samostatným aplikacím případně webovým formulářům, které lze nyní jednoduše vytvářet pomocí Maplet builderu.

6.1 Základní popis Maple

6.1.1 Maple na webu

`www.maplesoft.com` na tomto webu se nalézá prezentace firmy Maplesoft - výrobce systému Maple. Každý uživatel se zde může zaregistrovat a získat tak přístup do *Application Center*, které obsahuje širokou knihovnu příkladů využití Maple.

`beta.mapleprimes.com` uživatelské diskuzní fórum.

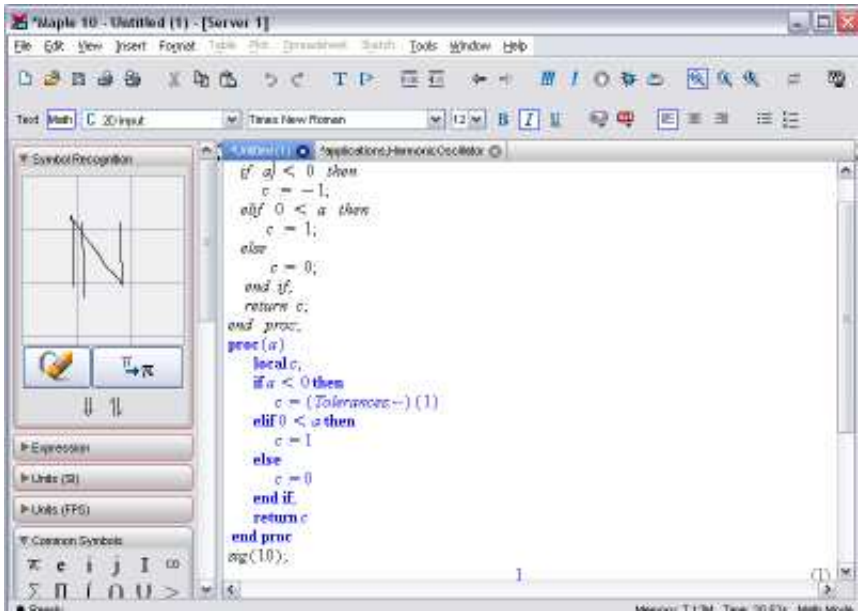
`www.maplesoft.cz` webová prezentace výhradního distributora Maple pro Českou a Slovenskou republiku. Obsahuje informace v češtině a dále spoustu zajímavých a užitečných odkazů na zdroje zabývajících se Maplem.

6.1.2 Prostředí

Prostředí systému Maple (viz obr. 6.3) je velmi intuitivní, přesto pro zadavání matematických výrazů je dobré znát základní příkazy, což výrazně urychlí práci.

Nejdůležitější klávesové zkratky

- Dokument v Maple obsahuje dva typy textu - klasického textu a matematického textu. Mezi těmito módy lze přepínat tlačítkem F5.
- Vyhodnocení zadaného matematického příkazu se provede stisknutím tlačítka Enter nebo Ctrl+= pro výsledek na stejné řádce.



Obrázek 6.3: Prostředí Maple 10

- K doplňování výrazů (například funkcí) lze použít kombinaci kláves Ctrl a Space.
- Nápověda Maple se vyvolá stisknutím tlačítek Ctrl+F1
- Rychlou referenci na zvolené funkci lze vyvolat tlačítky Ctrl+F2

Symbolické výpočty

Systém Maple užívá k interní reprezentaci veškerých objektů symboly, které mohou mít jeden z těchto významů:

čísla celá, racionální, reálná a komplexní, případně i algebraická

booleovské hodnoty pravda, nepravda, nevím

znaky písmena abecedy a další symboly

matematické objekty proměnné, matematické výrazy, rovnice a identity, posloupnosti, množiny, vektory, matice, polynomy a funkce jedné a více proměnných a jejich derivace a integrály, grafy funkcí jedné a dvou proměnných a jejich animace, systémy rovnic, nerovnic a algebraické struktury jako grupy, okruhy a algebry a jejich prvky, orientované grafy, apod.

datové struktury tabulky a datové soubory

vykonavatelé procedury funkce, grafy, algoritmy, atd.

Konečným cílem řešení matematického problému v Maple je vyjádření jeho řešení v explicitním analytickém tvaru nebo nalezení jeho symbolické aproximace. Pojmem algebraický se rozumí výpočty, které jsou prováděny přesně v souladu s pravidly algebry, namísto použití přibližné aritmetiky v pohyblivé řádové čárce, která způsobuje kvůli zaokrouhlování nepřesnosti ve výpočtu. Dalo by se říci, že Maple je naprosto přesný do doby než použijeme funkce z rodiny *eval*.

Příkladem těchto výpočtů jsou zjednodušování a úpravy matematických výrazů, derivování, rozklad polynomů, integrování funkcí a rozvoj funkcí v řady, analytické řešení rovnic, analytické řešení obyčejných i parciálních diferenciálních a integrálních rovnic, exaktní řešení systémů rovnic i nerovností atd.

Ačkoli mnoho standardních algebraických operací s matematickými výrazy je možno provádět jen s papírem a tužkou, tak u rozsáhlejších symbolických výpočtů začíná být nevýhodou větší délka vzorců a tím zdlouhavější práce matematika, který výrazy upravuje. Další nevýhodou těchto operací je nutné neustále maximální soustředění, aby se dosáhlo bezchybnosti algebraických operací a tím správnosti výsledku, zde tedy přichází do hry systémy počítačové algebry.

Další vlastností systému Maple je rozlišování velikosti písmen (angl. *case sensitive*). Maple tedy rozlišuje například mezi zápisem *pi*, *PI* a *Pi*. Zatímco první dva příklady reprezentují malé a velké písmeno řecké abecedy, třetí zápis reprezentuje konstantu π tedy hodnotu 3.141592654.

6.1.3 Základní příkazy

Po spuštění systému Maple se nahraje pouze jeho jádro (angl. *kernel*), které obsahuje tři části: interpret jazyka Maple, algoritmy pro numerické výpočty a funkce a procedury pro zobrazení výsledků a vstupní a výstupní operace. Další matematické funkce a operace jsou dále umístěny do knihoven, které dělíme na 3 základní skupiny:

hlavní obsahuje nejčastější příkazy, které se nahrávají ihned při spuštění (nejsou ale součástí jádra Maple)

packages balíčky obsahující skupinu příkazů pro danou část matematiky (např. LinearAlgebra - pro výpočty a operace s maticemi, DEtools - pro práci s diferenciálními rovnicemi atd.). Před použitím jednotlivých příkazů je potřeba nahrát balíček do paměti. To lze udělatjedním z následujících příkazů:

- **with(package);** nahraje celý balíček do paměti, používá se na začátku dokumentu.
- **with(package,cmd);** nahraje příkaz cmd z package do paměti.
- **package[cmd](parametry);** nahraje příkaz cmd z balíčku package do paměti, jedná se přímo o použití příkazu.

uživatelské jsou tvořeny méně frekventovanými matematickými příkazy. Nahrání těchto příkazů se provádí tímto příkazem: **readlib(cmd);**, kde cmd je příkaz, který chceme nahrát.

Jméno proměnné může obsahovat libovolnou posloupnost malých a velkých písmen a číslic. Toto jméno nesmí začínat číslicí a obsahovat mezery. Některá jména jsou vyhrazená pro užití Maple (např. signum, proc, end, if, atd.) Do proměnné lze přiřadit jakýkoliv matematický objekt. Toto přířazení probíhá operátorem **:=**. Rychlou nápovědu k hledanému výrazu lze vyvolat pomocí operátorů **?** *výraz*.

Výstup v Maple lze potlačit zadáním dvojtečky na konec příkazové řádky. V opačném případě může být typ výstupu jednou z těchto možností:

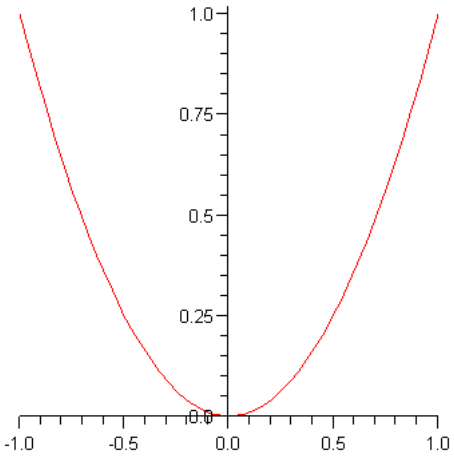
textový tento výstup je nejběžnější. Je výsledkem klasických operací.

```
> delka := 10;
```

$$10$$

grafický výstup zpracovaný formou grafu. (viz obr. 6.4)

```
> plot(x→x^2, -1..1)
```



Obrázek 6.4: Grafický výstup Maple

interaktivní výstup, který se stává průvodcem pro další interaktivní práci (viz obr. 6.5).

```
> dsolve[interactive](diff(x(t), t) = 3*x(t));
```

Máme-li výsledek v symbolickém tvaru jedná se sice o velmi přesné řešení, ale občas je pro naše účely vhodné zobrazit tento výsledek jako číslo. K tomu slouží funkce *evalf*. Můžeme si nechat vypsat například π s přesností 50 míst:

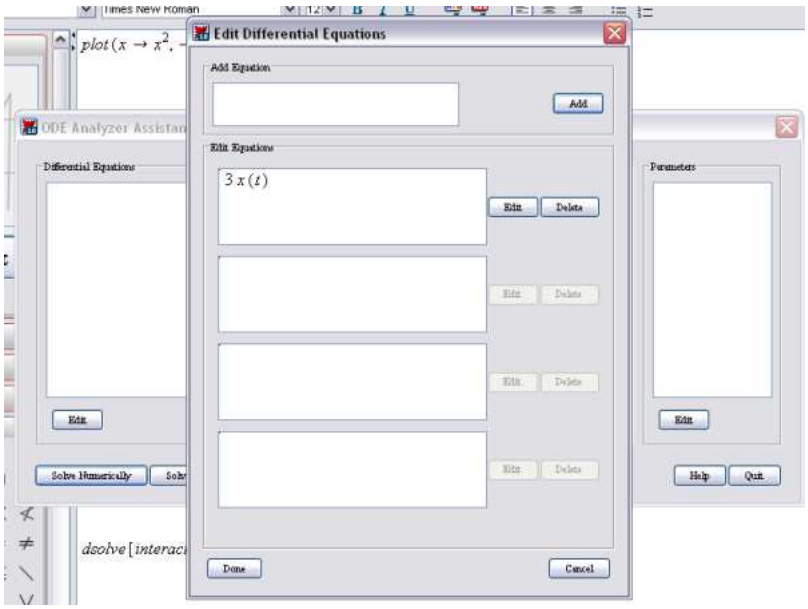
```
> evalf[50](Pi);
```

$$3.1415926535897932384626433832795028841971693993751$$

případně vyhodnotit integrál

```
> evalf(Int(tan(x),x=0..Pi/4));
```

$$0.3465735903$$



Obrázek 6.5: Interaktivní výstup Maple

6.1.4 Definice funkcí

Syntaxe pro definici funkce je následující:

```
nazev := proc(parametry)
    local [promenne]
    [prikazy]
end proc:
```

Pokud nejsou uvedeny žádné parametry, Maple použije pro zadané parametry interní označení args[i], kde i je pozice parametru. Číslování pozice probíhá od číslice 1 do $nargs$.

6.1.5 Řídící příkazy

if

Příkaz vyhodnotí podmínku a podle výsledku provede jednu z možných větví. Pokud je splněna *podmínka 1* provedou se *příkazy 1*, jinak se provedou *příkazy 3*. V syntaxi je též povolen neomezený počet *elif* větví - není-li splněna žádná předchozí podmínka a je splněna podmínka za klíčovým slovem *elif* provedou se příkazy následující za ní a celý řídící příkaz se ukončí. Řídící příkaz *if* má následující syntaxi:

if *podmínka 1* **then** *příkazy 1* [**elif** *podmínka 2* **then** *příkazy 2*] **else** *příkazy 3* **end if**

Příklad

```
> signum := proc(a)
    local c;
    if (a < 0) then
        c := -1;
```

```
    elif (a > 0) then
        c := 1;
    else
        c := 0;
    end if
    return c;
end proc:

> signum(10);
  signum(-23);
  signum(0);
```

1
-1
0

for

Syntaxe příkazu *for* je následující:

for *proměnná* **from** *odkud* [**by** *krok*] **to** *kam* [**while** *podmínka*] **do** [*příkazy*] **end do;**
nebo
for *proměnná* **in** *struktura* [**while** *podmínka*] **do** [*příkazy*] **end do;**

Příkaz *for* je konečným cyklem, kde jsou příkazy provedeny tolikrát, kolikrát je splněna podmínka „odkud - kam“ případně pro každý prvek ze zadané struktury. Omezení průběhu může být ještě dáno za klíčovým slovem *while*, které není povinné.

Příklad

```
integral := proc(cislo)
    local integral, tmp;
    integral := 1;
    for tmp from cislo by -1 to 1 do
        integral = integral * tmp;
    end do:
    return integral;
end proc:

> integral(4);
```

24

while

Příkaz *while* je hodně podobný předchozímu. Opět se jedná o cyklus, ale tentokrát s přesněji neurčeným počtem krokům a potenciálně nekonečným. Syntaxe příkazu je tato:

while *podmínka* **do** *příkazy* **end do:**

Dokud je splněna *podmínka* (vyhodnocuje se jako **true**) provádí se *příkazy*.

Jako příklad uvedeme převod čísla v desítkové soustavě do binární:

```
dec2bin := proc(par)
  local binarni, y, x:
  binarni := "":
  x := par;
  while x > 0 do
   y := x mod 2:
   binarni := cat(y, binarni):
   x := (x - y ) / 2:
  end do:
  return binarni;
end proc:
dec2bin(23);
```

$$10111$$

a pro ověření:

> **convert(23, binary);**

$$10111$$

6.1.6 Zjednodušování výrazů

Pokud neproběhne automatické zjednodušení výrazu a jsme přesvědčeni, že by nějaké zjednodušení bylo ještě možné (například u součtů), můžeme toto zjednodušení vynutit některým z následujících příkazů:

simplify Slouží pouze k zjednodušení výrazu. Jako druhý parametr můžeme zadat zjednodušení, které chceme provést, viz následující příklad:

> **simplify(sin(x)^2 + ln(2*x) + cos(x)^2);**

$$1 + \ln(2) + \ln(x)$$

> **simplify(sin(x)^2 + ln(2*x) + cos(x)^2, trig);**

$$1 + \ln(2\ x)$$

Dále lze k tomuto příkladu přidat ještě nějaký předpoklad pomocí slova **assume**:

> **g:=sqrt(x^2);**
> **simplify(g, assume=real);**

$$|x|$$

> **simplify(g, assume=positive);**

$$x$$

collect Tato funkce upraví svůj první parametr jako polynom podle druhého parametru:

```
> collect(a^3*x-x+a^3+a, x);
```

$$(a^3 - 1)x + a^3 + a$$

Funkce **collect** netřídí výstup, lze proto použít funkci **sort** a výsledek pak vypadá takto:

```
> sort(collect(a^3*x-x+a^3+a, x));
```

$$a^3 + a + (a^3 - 1)x$$

expand Tato funkce provede roznásobení vyšších mocnin polynomů, rozklad goniometrických funkcí atd.

```
> expand(sin(x+y));
```

$$\sin(x)\ \cos(y) + \cos(x)\ \sin(y)$$

combine Tato funkce aplikuje transformace integrálů, sum, mocnin a dalších výrazů do jednoho:

```
> combine(Int(x,x=a..b)-Int(x^2,x=a..b));
```

$$\int_a^b -x^2 + x dx$$

normal Funkce normalizuje výrazy - většinou zlomky:

```
> normal( (x^2-y^2)/(x-y)^3 );
```

$$\frac{y+x}{(-x+y)^2}$$

factor Funkce spočítá faktorizaci zadaného výrazu. Pokud je zadán druhý parametr provede rozklad na čtverec.

```
> factor(x^2-1);
```

$$(x-1)\ (x+1)$$

6.1.7 Grafy v Maple

Příkaz plot

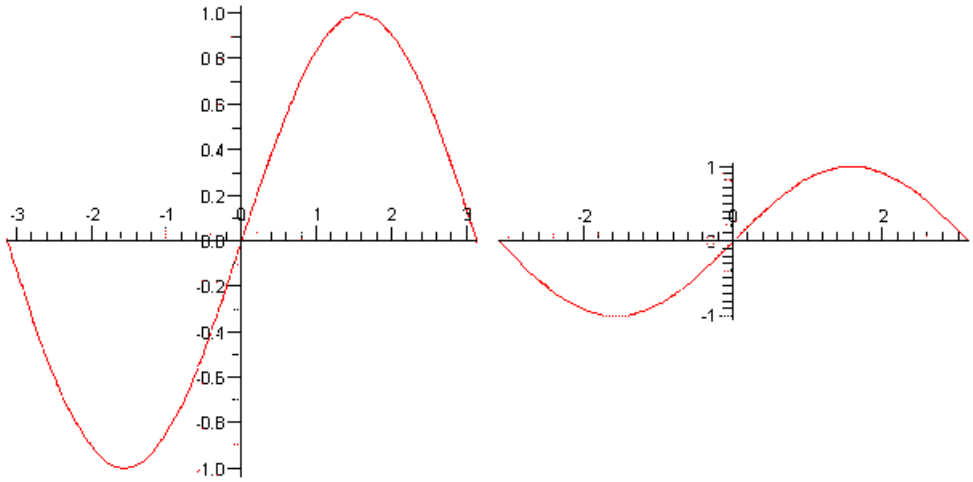
Základním příkazem pro zobrazování grafu v systému Maple je příkaz **plot**. Jeho syntaxe je:

```
> plot(výraz, rozsah)
```

Výhodou Maple 10 je možnost zapsat pouze výraz do dokumentu a pomocí pravého tlačítka rozbalíme kontextové menu a v něm vybereme nabídku „Plots“. Zde si již můžeme zvolit z nabízených možností (2D graf, 3D graf, graf pro implicitní vyjádření atd.)

Vytvořený graf již můžeme editovat pomocí kontextového menu, avšak pokud chceme takto upravený graf vidět při příštím spuštění (pokud si upravený graf neuložíme přímo do dokumentu, což ho podstatně zvětší) je potřeba použít rozšiřující vlastnosti příkazu **plot**. Tyto možnosti se zadávají za rozsah a jsou tedy dalším parametrem. Lze zadat následující vlastnosti:

scaling=constraing ovlivní poměr jednotek na osách grafu, aby byly v poměru 1:1. Tuto změnu je možné vidět na obrázku 6.6



Obrázek 6.6: Graf, jehož osy nemají (vlevo) a mají (vpravo) stejné měřítko.

axes upravuje zobrazení os

boxed zobrazí osy okolo grafu na všech stranách

frame zobrazí osu x dole a osu y vlevo

none nezobrazí žádné osy

normal implicitní nastavení

color nastaví barvu křivky v grafu

discont hodnota „true“ způsobí nezobrazení spojnic grafu v nespojité funkci. Pokud je funkce nespojitá Maple automaticky vytváří spojnici nejbližších bodů vlevo a vpravo od nespojitosti. Příklad pro funkci tan je zobrazen na obrázku 6.7

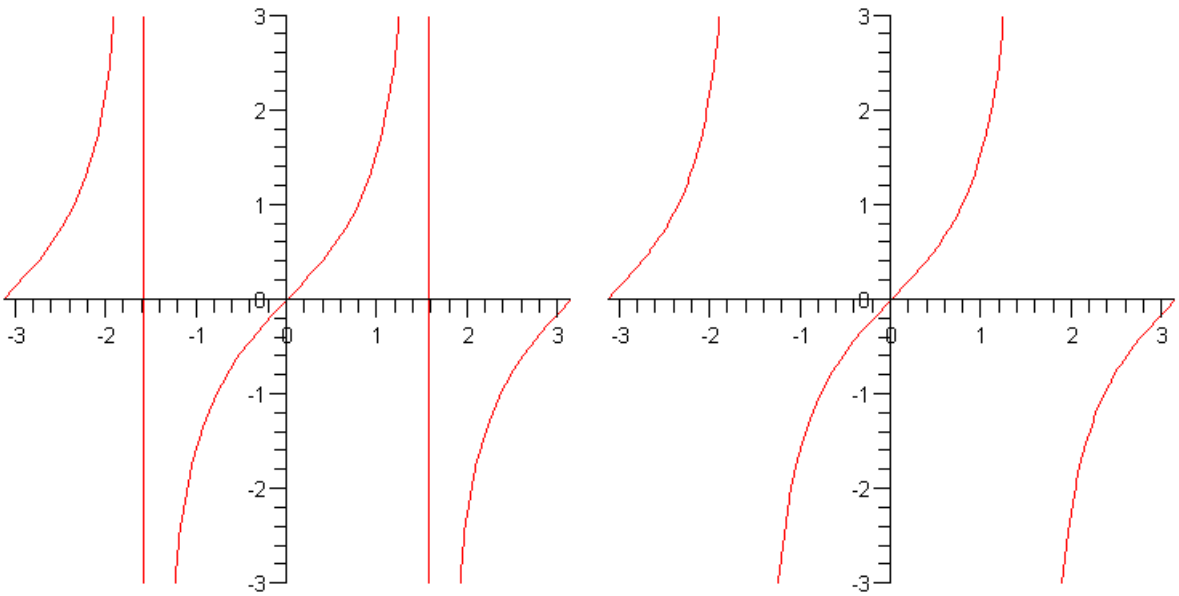
filled=true vyplní oblast ohraničenou křivkami zadanou barvou. Tato vlastnost je dostupná pouze u některých funkcí (plot, contourplot, implicitplot, listcontplot, polarplot a semilogplot.)

gridlines povoluje (*true*) nebo zakazuje (*false*) zobrazení mřížky v grafu

labels nastavuje popisky os

labeldirections nastavuje směr zobrazení popisku jednotlivých os. Povolené jsou hodnoty „horizontal, vertical“.

legend nastavuje legendu ke grafu. Pokud je zobrazeno více křivek, je nutné zadat tuto legendu pro každou křivku jako seznam popisů.



Obrázek 6.7: Graf se spojnicemi (vlevo) a bez spojnic (vpravo) pro **plot(tan, -Pi .. Pi, -3 .. 3, discont = true)**

linestyle změní styl vykreslované křivky („SOLID, DOT, DASH, DASHDOT“ - normální, tečkovaná, čárkovaná, čerchovaná)

resolution nastavuje rozlišení na ose X. Defaultní hodnota je 200. Větší číslo vede k přesnější křivce, ale zvyšuje se výpočetní náročnost a čas spotřebovaný pro vykreslení grafu.

thickness změní tloušťku vykreslované křivky.

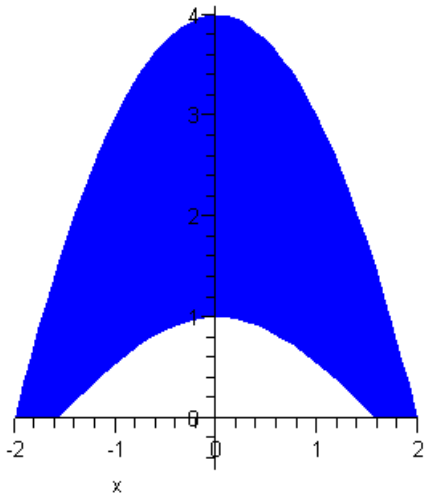
title zobrazí nadpis grafu.

Grafy je možné i sjednocovat (viz obr. 6.8. Lze toho docílit zadáním seznamu nebo množiny výrazů místo výrazu do definice funkce případně opět zapsáním pouze tohoto seznamu nebo množiny výrazů a použití kontextového menu. Parametry pro jednotlivé křivky pak zadáváme jako seznam parametrů:

```
> plot([cos(x), sin(x)], x=0..Pi, color=[red, blue], thickness=[2, 3]);
```

Mnohem více možností však nabízí příkaz **display** z balíku **plots**, který dovoluje vložení libovolné grafové struktury (například textplot, implicitplot). Parametry grafu stačí zadat jednou a není třeba je zadávat několikrát v seznamech jako v předchozím případě. Jesliže jsou některé parametry specifické pro jednotlivé grafy, zadávají se při vytvoření těchto grafů.

Pro vytvoření trojrozměrných grafů lze použít funkci **plot3d**, která má velmi obdobné parametry jako příkaz **plot**. Příklad výstupu této funkce je zobrazen na obrázku 6.9



Obrázek 6.8: Graf oblasti **plot([cos(x), 4-x^2], x = -2 .. 2, filled = true, color = ([white, blue]), scaling = constrained)**

Balík plots

Tento balík obsahuje několik dalších typů grafu. Pomocí příkazu **display** je pak umí všechny vykreslit do jednoho jediného grafu. Následuje výpis těchto funkcí:

fieldplot vytvoří graf z pole vektorů

complexplot vytvoří graf v komplexním oboru hodnot

contourplot vytvoří graf včetně jeho vrstevnic

coordplot vytvoří graf v jiném systému souřadnic. Dostupné jsou následující systémy: bipolar, cardioid, cartesian, cassinian, elliptic, hyperbolic, invcassinian, invelliptic, logarithmic, logcosh, maxwell, parabolic, polar, rose a tangent.

listdensityplot vytvoří graf hustoty zadaných hodnot

implicitplot vytvoří graf pro funkce zadané v implicitním tvaru (např. rovnice elipsy)

listplot vytvoří graf ze zadaných bodů a ty spojí

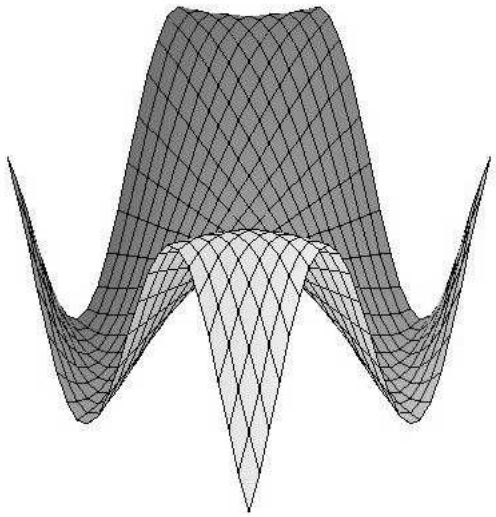
matrixplot vytvoří graf ze zadané matice, sloupce a řádky matice jsou považovány za jednotku na ose a hodnota v matici je vynesena na osu z.

odeplot vykresluje numerické řešení funkce **dsolve** pro řešení diferenciálních rovnic

pointplot vytvoří graf ze zadaných bodů

polarplot vytvoří graf v polárních souřadnicích

polygonplot vytvoří mnohoúhelník ze zadaných bodů



Obrázek 6.9: 3D graf **plot3d(sin(x*y), x = -2 .. 2, y = -2 .. 2)**

polyhedraplot vytvoří mnohostěn ze zadaných bodů

spacecurve vytvoří trojrozměrnou křivku

sparsematrixplot vytvoří graf rozložení nenulových hodnot v řídkých maticích. Řádky a sloupce jsou brány jako osy x a y.

surfdata vytvoří plochu křivky ze zadaných bodů

textplot vytvoří textový graf - používá se pro přidání textových popisků k jiným grafům

tubeplot vytvoří trubicový obal okolo zadané křivky

Pro většinu z těchto funkcí existuje i varianta pro trojrozměrné grafy - jméno tohoto příkazu lze odvodit přidáním „3d“ za uvedený název.

6.1.8 Řešení diferenciálních rovnic

Během modelování biologických problémů máme často potřebu řešit diferenciální rovnici nebo systém diferenciálních rovnic. Maple poskytuje k řešení diferenciálních rovnic příkaz **dsolve** a balíček příkazů **DEtools**. Příkaz **dsolve** obsahuje mnoho volitelných argumentů, pomocí kterých vyřešíme rozličné druhy diferenciálních rovnic. Nyní se omezíme na jeho základní použití při řešení jednoduchých diferenciálních rovnic v obecném tvaru i s počátečními podmínkami (Cauchyova počátečního problému).

Řešení nehomogenní lineární rovnice prvního řadu obecně a s předepsanou počáteční podmínkou:

```
> linrov := sin(x)*diff(y(x),x)-cos(x)*y(x)=cos(x):
> dsolve(linrov);
```

$$\{y(x) = -1 + \_C1 \sin(x)\}$$

```
> dsolve(linrov,y(Pi/2)=0);
```

$$y(x) = -1 + \sin(x)$$

Řešení homogenní rovnice druhého řádu s předepsanými počátečními podmínkami:

```
> difrov:=diff(y(x),x,x)+diff(y(x),x)-y(x)/x=0:
> dsolve({difrov,y(0)=0,D(y)(0)=1});
```

$$y(x) = x$$

Další specifické možnosti a nastavení parametrů lze nalézt v nápovědě. Diferenciální rovnice je možné řešit interaktivně pomocí příkazu **dsolve[interactive]**.

Z balíčku **DEtools** jsou velmi užitečné funkce **DEplot, DEplot3d** pro vykreslení řešení diferenciální rovnice nebo jejich systému a příkaz **odeadvisor** pro klasifikaci typu zadané rovnice, díky kterému můžeme rozhodnout o metodě jejího numerického nebo symbolického řešení. K ověření správného výpočtu slouží příkaz **odetest**, jenž na základě dvou parametrů (řešení, zadání) rozhodne, zda je řešení správné. Pokud ano, vrací nulovou hodnotu a v opačném případě nenulovou.

6.2 Novinky v Maple 10

6.2.1 Nový formát dokumentu

Nová verze Maple 10 obsahuje zcela nový typ dokumentu (viz obr. 6.1), který je označován jako tzv. Rich Technical Dokument (RTD). Systém Maple se stal velmi intuitivní a pomocí interaktivní kontextové nabídky je uživatel schopen jen s pomocí základních počítačových dovedností vytvářet plně interaktivní a komplexní dokumenty, které mohou sloužit dokonce jako výstup technických aplikací, popř. jako dokumentace. Žádném obdobný CAS systém (Mathematica, MathCad, Derive, atd.) takovouto vlastnost neobsahuje a proto je tedy Maple v tomto ohledu zcela revoluční. Jeho inovace přináší zlepšení možností využití celého systému nejen pro pokročilé uživatele, ale zejména pro začínající studenty. Tradiční zápisník je uživatelům stále k dispozici, ale neumožňuje jednoduše vytvářet plně interaktivní dokumenty.

6.2.2 Rozpoznávání symbolů

Další užitečnou pomůckou je nástroj pro rozpoznávání symbolů (viz obr. 6.2 vpravo). Na pracovní ploše Maple se implicitně zobrazuje v levém sloupci s nástroji. Pokud uživatel nemůže nalézt některý speciální znak nebo nezná-li jeho přesný zápis, může zkusit pomocí myši tento znak napsat a potom nechat Maple pokusit se rozpoznat tento znak. Možné varianty pak jsou zobrazeny v liště pod napsaným textem. Vybraný symbol lze ihned vložit do vytvářeného dokumentu.

6.2.3 Kontextové menu

Maple provádí uživatele na každém kroku přizpůsobeným kontextovým menu. Toto menu obsahuje vždy aktuálně dostupnou funkcionalitu k danému matematickému objektu. Grafy

umožňuje upravovat, otáčet, měnit souřadnice nebo exportovat do různých grafických formátů. Diferenciální rovnice naopak umožnuje řešit symbolicky, interaktivně nebo numericky, zároveň nabízí export do různých formátů. Kontextové menu pro diferenciální rovnici lze vidět na obrázku 6.2 uprostřed.

6.2.4 Nové packages

AudioTools poskytuje nástroje pro čtení a zápis do audio formátu wave.

DocumentTools je kolekci příkazů, které umožňují programově přistupovat k interaktivním komponentám v dokumentu.

ImageTools poskytují příkazy pro práci s rastrovými obrázky formátů *.jpeg, *.tiff, *.bmp a pomocí této knihovny lze provádět i základní obrazové operace, jako např. konvoluce apod.

IntegrationTools je balíček příkazů, které umožňují získat jednotlivé části počítaných integrálů:

```
> with(IntegrationTools):
> i:=Int(x*exp(x), x=-5..5);
```

$$\int_{-5}^{5} x e^x \, dx$$

```
> GetIntegrand(i);
```

$$x e^x$$

```
> GetRange(i);
```

$$-5..5$$

ProcessControl poskytuje příkazy pro tvorbu různých statistických grafů včetně výpočtů mezních hodnot.

RegularChains Tento balíček je určen pro řešení algebraických rovnic a studium jejich řešení.

Statistics je vytvořen pro statistické výpočty a obsahuje přes 35 příkazů, nahrazuje původní knihovnu stats, ale není s ní kompatibilní. Tento balíček obsahuje i interaktivní průvodce.

Student Tomuto balíku bude věnována speciální část. Jedná se o balík pro podporu výuky.

Units jsou určeny pro práci s jednotkami. Je možné používat jednotky SI a FPS. Balík umí automaticky upravovat jednotky, jak uvádí následující příklad:

```
> with(Units[Standard]):
> delka := 100 Unit(m):
> cas := 5 Unit(s):
> hmotnost := 20 Unit(kg):
> sila := (hmotnost*delka)/cas^2;
```

$$80\ \mathrm{Unit(N)}$$

Tolerances je určen pro výpočty s tolerancemi.

```
> with(Tolerances):
> a := 2 &+- 0.1;
```

$$a := (2.00) \pm (0.100)$$

```
> b := 3 &+- 0.05;
```

$$b := 3.00 \pm 0.0500$$

```
> a+b;
```

$$5.00 \pm 0.150$$

```
> a*b;
```

$$6.00 \pm 0.400$$

```
> a^b;
```

$$8.13 \pm 1.48$$

```
> sin(a);
```

$$0.905 \pm 0.0415$$

```
> exp(1/b);
```

$$1.40 \pm 0.00776$$

Typesetting je určen zejména pro ovlivnění sazby v Maple pomocí příkazového řádku

6.3 Student

Systém Maple poskytuje široké možnosti pro podporu výuky matematiky a dále na ni navazujících věd. Systém Maple je vhodný pro výuku matematiky zejména proto, že obsahuje vizuální nástroje, pomocí kterých je možné modelovat základní matematické problémy. Pro podporu výuky je nově vytvořen balík, který se jmenuje Student. Obsahuje následující části

Calculus1 funkce jedné proměnné

LinearAlgebra lineární algebra

MultivariateCalculus funkce více proměnných

Precalculus základní matematické problémy

VectorCalculus vektorový počet

Všechny tyto jeho části obsahují jak základní příkazy pro znázornění fundamentálních matematických problémů, tak i interaktivní průvodce a funkce pro grafické znázorňování těchto problémů.Každá tato část má tedy následující podčásti:

vizualizace Příkazy určené pro grafické znázorňování matematických problémů, které jsou určeny pro lepší pochopení dané problematiky, jsou základem této knihovny. Tyto příkazy mají obvykle grafický výstup, ale je možné pomocí parametru output tento výstup změnit (u některých příkazů toto nelze). Více informací je k dispozici v nápovědě systému Maple, kde lze nalézt také dokument s příklady pro tuto část knihovny.

interaktivní průvodci Každá knihovna jich obsahuje několik. Jsou postaveny na technologii Mapletů. Jejich funkčnost lze odvodit z jejich názvů, které výstižně popisují řešený problém.

výpočty krok za krokem Jde o systém, který je v určitých případech schopen pomáhat studentům při řešení jejich problémů. V nápovědě je k dispozici ukázkový soubor pro aplikaci těchto příkazů: Clear, GetMessage, GetNumProblems, GetProblem, Hint, Rule, Show, ShowIncomplete, ShowSteps, Understand, Undo, WhatProblem. Pokud jde o jejich použití, je velmi jednoduché a stačí se jen podívat do nápovědy systému Maple.

ostatní příkazy Další příkazy, které lze využít v souvislosti s obsahem dané knihovny.

6.4 Příklad využití Maple k odhadu parametrů

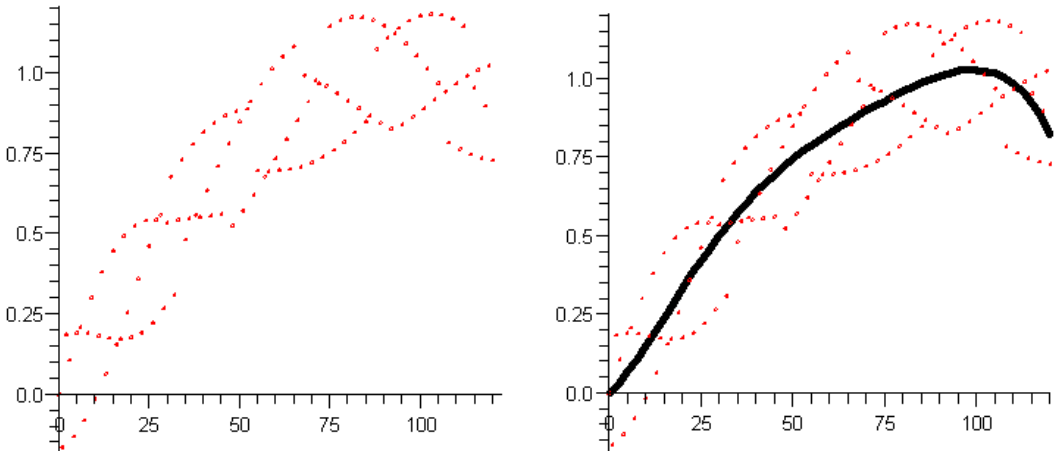
Jako příklad použijeme odhad parametrů pomocí metody nejmenších čtverců. Metoda nejmenších čtverců (angl. Least squares) je jedna z často využívaných metod pro nalezení aproximační funkce pro daná empiricky zjištěná data. Naměřená data mohou obsahovat chyby a proto nedokážeme nalézt křivku, která by přesně kopírovala tato data a měla námi požadovaný tvar[1]. Naším cílem tedy je proložit těmito daty křivku, která bude co

[1] Tímto tvarem je myšlena jednoduchá křivka. Každou množinou bodů lze proložit křivku, která prochází všemi body - tato křivka je interpolační polynom.

nejvěrněji kopírovat naměřená data a minimalizuje tak odchylku od nich.

Pro každý bod $X = [x_i, y_i]$ určíme vzdálenost jeho funkční hodnoty $F(x_i)$ od hodnoty y_i. Tento diferenciál by mohl nabývat kladných a záporných hodnot a výsledná hodnota by tak nebyla vypovídající, proto se tento diferenciál umocňuje na druhou, čímž se zbavíme tohoto nedostatku. Suma těchto druhých mocnin diferenciálu určuje přesnost získané křivky. Jak je z názvu patrné budeme hledat minimální hodnotu. Tuto metodu lze použít i pro vícerozměrná data, my se však omezíme pouze na dvourozměrný výpočet. Prokládaná křivka může být lineární nebo nelineární. V případě hledání lineární křivky (přímky) je řešení jednoduché a proto se zaměříme na složitější případ, který se vyskytuje častěji i v reálném prostředí.

Obrázek 6.10 zobrazuje naměřená data a křivku nalezenou pomocí metody nejmenších čtverců.



Obrázek 6.10: Vlevo: naměřená data, vpravo: křivka získaná pomocí metody nejmenších čtverců

Za předpokladu, kdy známe rovnici křivky a potřebujeme znát pouze řešení parametrů se opět jedná o jednoduchý příklad. Co tedy dělat pokud neznáme ani rovnici křivky. V tomto případě je vhodné uvažovat nějaký polynom. Z praktických výpočtů se ukazálo, že nejvhodnějším polynomem je polynom stupně 6 případně 7. Polynom vyšších stupňů může vést k získání křivky podobné interpolačnímu polynomu, která není vyhovující.

Jak tedy postupovat v Maple si ukážeme na následujícím příkladu:

Nejprve získáme naměřená data. V našem případě si vygenerujeme „náhodná“ data[2] Tato data lze pozorovat na obrázku 6.10 vlevo.

```
> X:=[$0..120]:
> N:=nops(X):
> Y:=map(u -> evalf(sin(u/60)+sin(35.7*u)/5, 4), X);
```

Y := [0., -.1652, .1844, .1064, -.1314, .1913, .2080, -0.815e-1, .1891, .3006, -0.160e-1, .1821, .3807, 0.640e-1, .1746, .4454, .1556, .1711, .4934, .2554, .1758, .5247, .3589, .1919, .5403, .4614, .2219, .5427, .5584, .2671, .5353, .6773, .3085, .5421, .7318, .4815, .5469,

[2]Jak je patrné nejedná se o náhodná data, ale o data vytvořená se záměrnou chybou ze zadané křivky.

.7781, .5584, .5510, .8163, .6350, .5554, .8460, .7098, .5612, .8676, .7815, .5242, .8814, .8490, .5710, .8882, .9108, .6220, .6969, .9659, .6768, .6956, 1.013, .7342, .6968, 1.052, .7932, .7011, 1.083, .8525, .7091, .9929, .9111, .7215, .9781, .9673, .7383, .9594, 1.145, .7597, .9377, 1.164, .7854, .9143, 1.173, .8150, .8904, 1.173, .8478, .8672, 1.165, 1.074, .8459, 1.148, 1.110, .8273, 1.123, 1.140, .8408, 1.092, 1.163, .8641, 1.054, 1.178, .8894, 1.013, 1.184, .9160, .9680, 1.181, .9425, .7830, 1.168, .9676, .7627, 1.146, .9905, .7467, .9530, 1.009, .7353, .8977, 1.024, .7282]

Dále si musíme vytvořit obecný tvar námi požadovaného polynomu stupně 6.

```
> n:=6:
> F:=unapply(sum(a[i]*x^i, i=0..n), x);
```

$$\mathrm{F} := x \rightarrow a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6$$

A vytvoříme si pole všech proměnných složek:

```
> Var:={seq(a[j], j=0..n)};
```

$$\mathrm{Var} := \{a_0, a_1, a_2, a_3, a_4, a_5, a_6\}$$

Samotné řešení pak spočívá v provedení následujících kroků, kdy provedeme tyto operace:

Vypočítáme si sumu druhých mocnin diferenciálů:

```
> q:=expand(sum((Y[i]-F(X[i]))^2, i=1..N)):
```

Dále provedeme derivaci podle každé proměnné:

```
> Eq:=map(u -> diff(q, u), Var):
```

A vyřešíme:

```
> Sol:=solve(Eq,Var);
```

$$\mathrm{Sol} := \{a_6 = 1.992299970 \cdot 10^{-12}, a_5 = -1.170662919 \cdot 10^{-9}, a_4 = 2.210692171 \cdot 10^{-7}, a_3 = -0.00001807107103, a_2 = 0.0005584109188, a_1 = 0.01121230479, a_0 = 0.001485715047\}$$

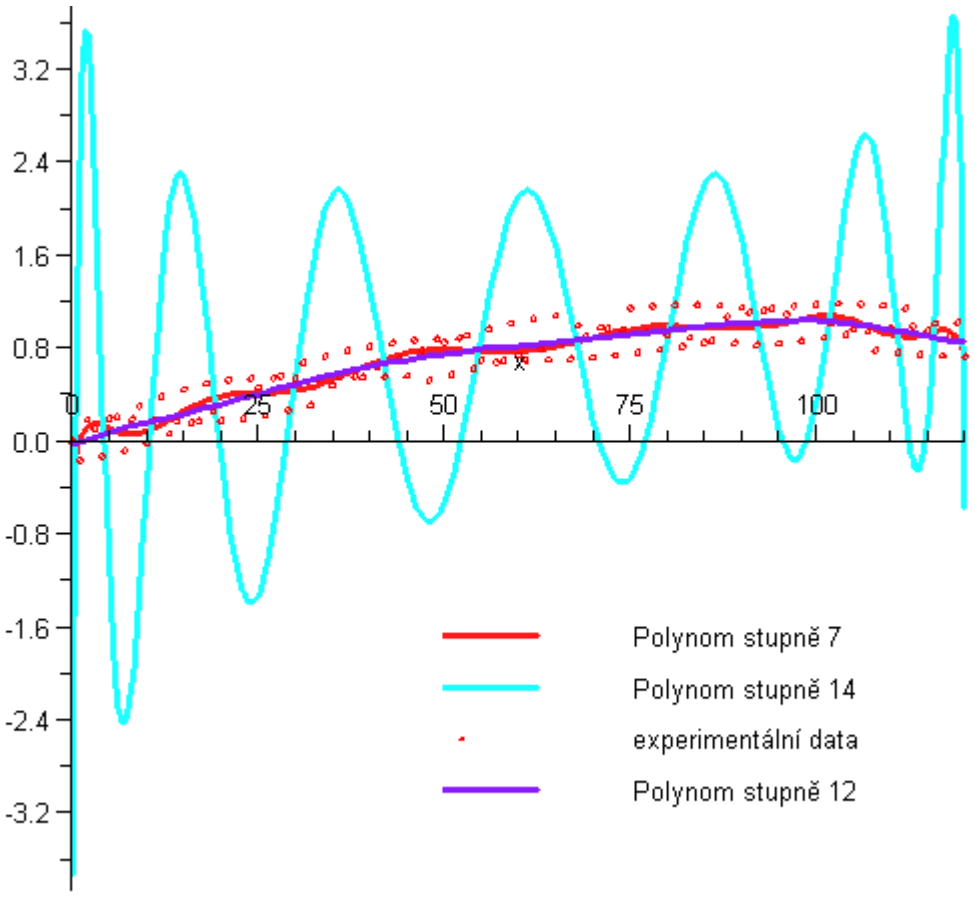
Na závěr provedeme pouze substituci vyřešených parametrů do zvoleného polynomu:

```
> Fv:=subs(Sol, F(x));
```

$$\mathrm{Fv} := 0.001485715047 + 0.01121230479\,x + 0.0005584109188\,x^2 - 0.00001807107103\,x^3 + 0.0000002210692171\,x^4 - 0.000000001170662919\,x^5 + 1.992299970 \times 10^{-12}\,x^6$$

Křivku získanou metodou nejmenších čtverců pro vstupní data lze pozorovat na obrázku 6.10 vpravo.

Obrázek 6.11 ukazuje několik druhů křivek pro polynomy různého stupně (7, 12, 14). Jak je z obrázku patrné polynom stupně 14 již porušuje naši představu tvaru prokládané křivky.



Obrázek 6.11: Porovnání několika křivek získaných metodou nejmenších čtverců

Příloha A

Elektronické zdroje

A.1 České

A.1.1 Encyklopedie

http://cs.wikipedia.org/
   http://www.mozek.cz/
   http://www.vseved.cz/
   http://www.cojeco.cz/
   http://www.navajo.cz/
   http://encyklopedie.seznam.cz/
   http://www.cojeco.cz/
   http://www.coto.je/

A.1.2 Vyhledávače

http://www.portalstm.cz/
   http://www.medvik.cz/medvik/

A.1.3 Specializované weby o biologii a medicíně

http://www.gate2biotech.com/cz/
   http://www.mediclub.cz/

A.2 Anglické

A.2.1 Encyklopedie

Obecné

http://www.wikipedia.org/
   http://encarta.msn.com/
   http://www.britannica.com/
   http://www.answers.com/
   http://www.thefreedictionary.com/

http://acronyms.thefreedictionary.com/
http://www.encyclopedia.com/
http://www.about.com/
http://www.questia.com/

Technické

http://www.techweb.com/encyclopedia/
http://whatis.techtarget.com/

Matematické

http://www.mathacademy.com/pr/prime/index.asp
http://mathworld.wolfram.com/
http://planetmath.org/encyclopedia/
http://math.about.com/
http://www.math.com/

Specializované weby o biologii a medicínu

http://www.biology-online.org/
http://users.rcn.com/jkimball.ma.ultranet/BiologyPages/
http://biology.about.com/
http://mrb.niddk.nih.gov/sherman/gallery/ - galerie modelu
http://www.ncbi.nlm.nih.gov/entrez/

A.2.2 Vyhledávače a rozcestníky

http://scholar.google.com/
http://www.scirus.com/
http://www.sc.edu/library/science/sciref.html
http://www.healthlinks.net/

A.2.3 Vyhledání definice

http://www.google.com/search?q=define:

Literatura

[1] B. Barnes a G. Fulford. *Mathematical Modelling with Case Studies: A Differential Equation Approach Using Maple.* T&F STM, 2002.

[2] V. Capasso a D. Bakstein. *An Introduction to Continuous-Time Stochastic Processes Theory, Models, and Applications to Finance, Biology, and Medicine.* New York: Springer, 2005.

[3] W. Gander a J. Hřebíček. *Solving Problems in Scientific Computing Using Maple and MATLAB.* Springer-Verlag Berlin, 4th edition, 2004.

[4] A. Bellouquid a M. Delitala. *Mathematical Modeling of Komplex Biological Systems A Kinetic Theory Approach.* Birkhauser Boston, 2006.

[5] A. Swishchuk a Jianhong Wu. *Evolution of Biological Systems in Random Media: Limit Theorems and Stability.* Springer, 2003.

[6] A. Saltelli, S. Tarantola, F. Campolongo, a M. Ratto. *Sensitivity Analysis in Practice: A Guide to Assessing Scientific Models.* Wiley, 2004.

[7] A.K. Konopka a M.J. Crabbe. *Compact Handbook of Computational Biology.* Marcel Dekker, 2004.

[8] Jones D.S. a Sleeman B.D. *Differential Equations and Mathematical Biology.* Chapman & Hall/CRC, 2003.

[9] J. Kalas a Z. Pospíšil. *Spojité modely v biologii.* MUNI, 2001.

[10] B. Haubold a T. Wiehe. *Introduction to Computational Biology: An Evolutionary Approach.* Birkhauser, 2006.

[11] J. Herod, R. Shonkwiler, a E. Yeargers. *An Introduction to the Mathematics of Biology.* Birkhauser, 1996.

[12] Larsen R. J. a Marx M. L. *An Introduction to Mathematical Statistics and Its Applications.* Prentice Hall, 3rd edition, 2001.

[13] Z. Pospíšil. Odhad parametrů logistické rovnice. *Biometrické metody a modely v současné vědě a výzkumu*, pages 179–187, 2002.

[14] A. Saltelli, K. Chan, a E. M. Scott. *Sensitivity Analysis.* Wiley, 2000.

[15] L. Pachter a B. Sturmfels. *Algebraic Statistics for Computational Biology.* Cambridge University Press, 2005.

[16] N.F. Britton. *Essential Mathematical Biology.* Springer, 2005.

[17] Murray J.D. *Mathematical Biology: I. An Introduction.* Springer, 3rd edition, 2004.

[18] P. Checkland a J. Poulter. *Learning For Action: A Short Definitive Account of Soft Systems Methodology, and its use Practitioners, Teachers and Students.* John Wiley & Sons, London, 2006.

[19] G.M. Jenkins a P.V. Youle. *Systems Engineering: A Unifying Approach in Industry and Society.* C.A.Watts & Co Ltd, London, 1971.

[20] Thieme H.R. *Mathematics in Population Biology.* Prentice Hall, 2003.

[21] F. Brauer a C. Castillo-Chavez. *Mathematical Models in Population Biology and Epidemiology.* Springer, 2001.

[22] T. Aven a U. Jensen. *Stochastic Models in Reliability.* Springer, 1999.

[23] Y. Takeuchi. *Global Dynamical Properties of Lotka-Volterra Systems.* World Scientific Publishing Company, 1996.

[24] G.F. Gause. *The struggle for existence.* Baltimore, 1934.

[25] A. Lesk. *An Introduction To Bioinformatics.* Oxford University Press, 2005.

[26] S. Asmussen a P.W. Glynn. *Stochastic Simulation: Algorithms and Analysis.* Springer, 2006.

[27] H.J. Kushner a P.G. Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time.* Springer, 2001.