evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
UNIVERSITAS MASARYKIANA BRUNENSIS
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Úvod do matematického modelování s využitím Maple

Jiří Hřebíček, Zdeněk Pospíšil, Jaroslav Urbánek

Červenec 2010

Příprava a vydání těchto učebních textů byly podporovány projektem ESF č. CZ.1.07/2.2.00/07.0318 **„Víceoborová inovace studia Matematické biologie“** a státním rozpočtem České republiky.

Předmluva

Publikace „*Úvod do matematického modelování s využitím Maple*“ vznikla v souvislosti s řešením projektu ESF č. CZ.1.07/2.2.00/07.0318 „VÍCEOBOROVÁ INOVACE STUDIA MATEMATICKÉ BIOLOGIE“, který si klade za cíl inovovat náplň a provázanost předmětů z kmenového souboru předmětů oboru „Matematická biologie“, studijního programu „Biologie“ Přírodovědecké fakulty MU, profilujících studijní obor a zajišťovaných pracovníky IBA MU.

Oproti koncepci až dosud používaného učebního textu: „*Jiří Hřebíček, Michal Škrdla: Úvod do matematického modelování*“ zaměřeného spíše na obecné metody matematického modelování, je předkládaná monografie orientována více do oblasti praktického řešení biologických modelů pomocí systému počítačové algebry Maple, včetně řešení problematiky neurčitosti a citlivosti parametrů modelu.

Publikace přináší vhodné postupy a metody pro řešení vybraných biologických modelů. Je možné ji využívat ve studiu matematického modelování dvěma způsoby. Studentům, kteří mají větší matematické a informatické znalosti (například v oboru „Matematická biologie“, případně dalších studijních oborů a programů MU) bude pomáhat v tom, jak používat a vytvářet modely v systému počítačové algebry Maple. K tomu jim budou sloužit všechny kapitoly publikace. U studentů s nižšími matematickými a informatickými vědomostmi (typicky studenti dalších biologických či zdravotnických oborů, případně dalších programů MU) nebude třeba studovat kapitolu popisují systém Maple. Bude jim stačit nastudovat demonstrativní praktické příklady v systému Maple k lepšímu pochopení metodiky matematického modelování.

Vytvořený text pomůže zkvalitnit výuku jednoho z kmenových předmětů oboru „Matematická biologie“ pomocí vybraných speciálních příkladů řešení problémů matematického modelování, kde se využívá nových e-learningových vlastností Maple. To umožní používat tuto publikaci rovněž vysokoškolským učitelům ke zlepšení jejich pedagogické dovednosti pomocí efektivního využívání systému Maple při výuce řešení ilustrativních příkladů matematického modelování v biologii.

V Brně 30. dubna 2010

Jiří Hřebíček
Zdeněk Pospíšil
Jaroslav Urbánek

ISBN 978-80-7204-691-1

1 Úvod do matematického modelování a jeho členění

Matematické modelování proniklo do různých oborů přírodních, technických, ekonomických i sociálních věd a stalo se důležitým nástrojem při modelování a simulacích systémů, analýzách a předvídání různých procesů, jevů, chování druhů a stavů společenstev apod. V dalším textu se zaměříme na matematické modelování systémů, kde systémy budeme chápat jako určité abstrakce reálného světa (objektivní reality), které si lidé vytvářejí v procesu jeho poznání. Jako systém budeme zjednodušeně uvažovat (následující výčet je neúplný) např.:

a) *Proces*, *komplex procesů*, (např. pohyb kyvadla, tok elektrického proudu v obvodu, rozmnožování buněk a organismů, apod.), jímž rozumíme zákonité, na sebe navazující a vnitřně propojené změny nějakého objektu. Proces lze často číselně vyjádřit časovým průběhem nějaké hodnoty, resp. skupin hodnot. Můžeme dávat přednost obecnějšímu vyjádření, kde místo čísel vystupují prvky nějaké množiny. Pak *systémem* nazýváme každý objekt (konkrétní nebo abstraktní), jenž vstupnímu procesu určitého typu přiřazuje výstupní proces téhož nebo jiného typu. Toto přiřazení, které popisuje reakce výstupů na vstupy, se nazývá *chováním systému*, a proto se tato definice nazývá *behavioristická* (behaviour - angl. chování). Místo slova „přiřazení“ často používáme i slovo „transformace“, jenž je mnohdy z praktického hlediska výstižnější, protože mnoho systémů opravdu transformuje vstupní procesy na výstupní, tj. přetváří je.

b) Takový *objekt* (přirozený či umělý), který v každém časovém okamžiku má na vstupu nějaký vstupní prvek, na výstupu nějaký výstupní prvek a kromě toho je vždy v nějakém vnitřním stavu, přičemž jsou dány jednoznačné závislosti

   - stávajícího výstupního prvku na stávajícím stavu a vstupním prvku,
   - následujícího stavu na stávajícím stavu a vstupním prvku.

   Tato definice se nazývá *stavová* a je ekvivalentní s první definicí, tj. každý objekt vyhovující definici první vyhovuje i definici druhé a naopak.

c) *Soubor nějakých prvků a vazeb mezi nimi*, např. soužití dravce a jeho kořisti; výroba jeřábu s předepsanou nosností, minimalizace spotřeby vozidla na danou vzdálenost apod. S používáním této definice však nastávají určité potíže. Není snadné interpretovat slovo „vazba“, a ne každý objekt, který lze intuitivně zcela zřejmě považovat za systém, je komponován z několika jasně odlišitelných prvků.

d) *Soubor informačních, regulačních a řídících činností* vztahujících se k a) – c), např.: informační systém, řídící systém, komunikační systém, regulační systém.

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

f) Abstraktní myšlenkovou konstrukci atd. vytvářenou bez přímého vztahu k a) – d).

Použití matematického modelu systému přináší řadu výhod:

- Umožňuje zjistit informace o chování systému, i když z skutečného systému je to nemožné nebo obtížné.
- Urychluje proces poznání objektivní reality. Procesy, které ve skutečném systému probíhají pozvolna a dlouhodobě, lze pomocí modelu sledovat zrychleně během simulace (výpočtu), která závisí na použité informační a komunikační technologii (ICT).
- 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.
- Umožňuje variantní řešení, tj. simulaci a propočet celé řady variant možných výsledků řešení.
- Identifikuje vznik chybného poznání objektivní reality (na rozdíl od experimentu v reálném systému).

Modelování bývá většinou pojímáno jako transdisciplinární činnost, neboť se na něm mohou podílet poznatky z matematiky a fyziky, teorie systémů, teorie pravděpodobnosti, informatiky, kybernetiky či kognitivních věd, operačního výzkumu a jiných. Matematické modelování získává v posledních letech velký význam v praxi, ale také ve výuce studentů na vysokých školách přírodovědného, technického i ekonomického zaměření. Do matematického modelování, stejně jako i do jiných odvětví vědy, proniklo již od šedesátých let minulého století využití informačních a komunikačních technologií (ICT). Nyní si bez jejich využití neumíme matematické modelování představit. Požadavky na tyto počítačové aplikace — symbolické a numerické výpočty, na jejich vysokou přesnost, vizualizaci a interaktivní komunikaci s řešitelem atd. — vedly k vytvoření komplexních aplikačních programů jako jsou např. komerční programy: Matlab od firmy Mathworks Inc.[1], Maple od Maplesoft Inc.[2], MathCAD od MathSoft Inc.[3], Mathematica od Wolfram Reasearch, Inc.[4], MuPAD[5] vyvíjený universitou Paderhorne a firmou SciFace Software GmbH[6] atd. Podrobnější informace lze nalézt na webu[7].

Z volně přístupných (open source) programů zmíníme např. programy: MAXIMA pro symbolické výpočty[8], OCTAVE pro numerické výpočty[9] a R pro statistické výpočty[10]. Tyto aplikační programy lze využít v matematickém modelování během celého vývojového procesu, tj. identifikace, analýzy, vývoje, implementace, řešení a ověřování, případně modifikace matematického modelu. Volně přístupné zdroje odborné literatury k této problematice lze nalézt na webu[11].

[1] http://www.mathworks.com
[2] http://www.maplesoft.com
[3] http://www.mathsoft.com
[4] http://www.wolfram.com
[5] http://research.mupad.de/
[6] http://www.sciface.com/
[7] http://www.symbolicnet.org/www-sites.html
[8] http://maxima.sourceforge.net/
[9] http://www.gnu.org/software/octave/index.html
[10] http://www.r-project.org/
[11] http://www.symbolicnet.org/lit.html

V současné době se používají některé výše uvedené systémy (např. Maple, Matlab a Mathematica) na vysokých školách pro demonstraci probírané látky na cvičeních, případně jsou využívány studenty při individuální přípravě, analýze a řešení problémů, které jim zadávají jejich učitelé nebo vyplývajících z jejich závěrečných prací.

Charakteristickým rysem inovace výuky matematického modelování ve studijních programech vysokých škol v České i Slovenské republice, Evropské unii a v zemích OECD se v posledních deseti letech stává používání nových ICT (např. Grid Computing[12], Cloud Computing[13]) v rámci budování nového vědního oboru „Computational Science“ a „Mathematical Modelling“ [7].

Cílem stále více vysokých škol je ovšem zařadit do výuky předmět seznamující studenty s prací ve výše uvedených systémech, na které mají zakoupenu licenci, a využít jejich možností při návrhu, analýze, řešení a testování netriviálních matematických modelů. To je rovněž cílem této publikace, kde nejprve uvedeme, co je matematický model a metodologie matematického modelování. Na praktických příkladech ukážeme využití systému Maple pro matematické modelování.

Přístup k matematickému modelování s využitím ICT lze rozdělit do „black-box“ modelování[14], „white-box“ modelování a „shadow-box“ modelování, podle toho, v jakém rozsahu jsou předem známy informace o systému a způsobu jeho řešení.

Black-box model je systém, o kterém není známa a priori informace. White-box model je systém, kde jsou známy všechny potřebné informace. Prakticky všechny známé způsoby matematického modelování s využitím ICT náleží mezi black-box a white-box řešení modelů. Nedávno bylo zavedeno tzv. shadow-box modelování, kdy uživatel využívá jako základní black-box modelování a může si zvolit alternativně white-box modelování při řešení modelu systému. Toto umožňuje právě systém Maple, který je v tomto směru nejvíce rozvinut.

Obvykle je lepší používat co nejvíce a priori informací, pokud je to možné, a vytvořit mnohem přesnější matematický model systému. Tudíž white-box modely jsou obvykle považovány za vhodnější, protože pokud se použily informace o systému správně, pak model i jeho řešení se budou chovat správně. Často je apriorní informace dána v podobě znalosti typu funkcí týkající se jejich různých proměnných. Například, když chceme vytvořit model, jak lék funguje v lidském organismu (systému), víme, že obvykle množství léku v krvi v čase je exponenciálně klesající funkce. Víme však, že jsou zde ještě další neznámé parametry určující, jak rychle se množství léku v krvi vstřebá, a jaké je původní množství léku v krvi. Tento příklad tedy není typu white-box model. Jeho parametry musí být odhadnuty prostřednictvím nějakých jiných lékařských metod, a teprve poté je možné použít model s exponenciálně klesající funkcí.

1.1 Matematický model

Matematický model je abstraktní model[15], který využívá matematického jazyka k popisu chování systému. Používá se převážně v přírodních (fyzika, biologie, chemie apod.) a technických (strojírenství, elektrotechnika, stavebnictví apod.) vědách, ale také ve společenských vědách (ekonomie, sociologie, politické vědy apod.). Matematický model transformuje systém do matematického zápisu, který má následující výhody:

[12] `http://en.wikipedia.org/wiki/Grid_computing`

[13] `http://en.wikipedia.org/wiki/Cloud_computing`

[14] `http://en.wikipedia.org/wiki/Black_box_(systems)`

[15] Abstraktní model (nebo konceptuální model) je teoretická konstrukce, která reprezentuje fyzikální, biologický nebo sociální či ekonomický proces. Sestává z množiny proměnných a množiny logických nebo kvantitativních vztahů mezi proměnnými.

- formalizaci zápisu danou historickým vývojem (v současné době je vyvinuta uznávaná mezinárodní standardizace),
- přesná pravidla pro manipulaci s matematickými symboly a strukturami,
- možnost využití ICT pro zpracování a řešení vytvořeného modelu.

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

1.1.1 Základní prvky matematického modelu

Matematický model obvykle popisuje systém s pomocí množiny vstupních a výstupních proměnných, dále parametrů a množiny rovnic, které určují stavy systému a vztahy mezi proměnnými a parametry. Hodnoty proměnných mohou být např. reálná nebo celá čísla, booleovské hodnoty nebo textové řetězce anebo složitější struktury. Proměnné reprezentují nějaké vlastnosti systému, např. výstupy měřených systémů často ve tvaru signálů, vzorkovaná data, hodnoty počitadla, výskyt dané události či jevu (ano/ne) apod. Na model se můžeme dívat také jako na množinu funkcí, která popisuje vztahy mezi různými proměnnými.

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

- proměnné a parametry,
- matematické struktury,
- řešení.

Z hlediska ICT můžeme proměnné a parametry v modelu uvažovat jako:

- **Identifikované (pojmenované) proměnné a parametry.** Identifikovaná proměnná nebo parametr představuje konkrétní vlastnost reálného objektu, pojmenovanou názvem a fyzikální jednotkou, v níž se měří.
  Příklady: x_k je výměra pšenice ozimé v hektarech, x_r produkce pšenice ozimé na parcele „U křížku“ v tunách, c_{ik} vzdálenost dodavatele D_i od spotřebitele S_k v kilometrech.
- **Neidentifikované (pomocné) proměnné a parametry.** Slouží pro formalizaci matematického zápisu, implementaci algoritmů apod. Obvykle se uvažují v bezrozměrných jednotkách.
  Příklad: U lineárního programování se zavádějí pomocné proměnné, které umožní změnit „nerovnice na rovnice“.
- **Nekontrolovatelné proměnné.** Představují procesy v systému, jejichž míry nelze zjistit (jedná se o další typ neurčitosti).
  Příklady: V modelech kilmatu „ad hoc“ jsou charakteristiky počasí nekontrolovatelné parametry nebo proměnné, protože nelze využít počtu pravděpodobnosti pro jejich popis. Velikost míry inflace v chaotických a nestandardních tržních podmínkách nelze popsat ani pomocí pravděpodobnosti ani pomocí fuzzy funkce.

1.2 Klasifikace matematických modelů

Během identifikace a analýzy 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 obsahují proměnné, jež jsou abstrakcí hledaných prvků systému a operátorů nad těmito proměnnými. Operátory mohou reprezentovat algebraické operace, funkce, funkcionály, diferenciální operátory atd. Pokud jsou operátory v matematickém modelu *lineární*, hovoříme o *lineárních modelech*, v opačném případě o *nelineárních modelech*.

Můžeme také uvažovat modely se soustředěnými (u homogenních modelů) a distribuovanými (u heterogenních modelů) parametry. Mezi těmito skupinami leží mnoho jiných 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, ale 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 narození 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 popisuje a analyzuje systém bez zřetele k jeho časovému vývoji. Zobrazení se týká zpravidla určitého časového intervalu (týden, měsíc, rok apod.).

- **Modely dynamické.** Model popisuje 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 zahrnutí faktoru času do dosud statického 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.
Příklad: U lineárního programování k pravé straně maticové rovnice $A \cdot x \leq b$ přidáme časový vektor $u(t)$.

- **Modely deterministické.** Všechny proměnné, parametry a funkce v modelu jsou deterministické (nenáhodné) veličiny nebo funkce.
- **Modely stochastické.** Alespoň jedna proměnná, parametr nebo funkce v modelu je náhodná veličina nebo náhodná funkce.
- **Fuzzy modely.** Některé proměnné a parametry jsou „fuzzy“ (nepřesně ohraničené, neostré, neurčité, vágní). Funkce příslušnosti ve fuzzy logice jim umožňuje přiřadit příslušnost k množinám v rozmezí od 0 do 1, včetně obou hraničních hodnot. Uplatnění fuzzy modelování je účelné ve všech případech, kdy se řeší problém spojený s neurčitostí, s nepřesností a musí se pracovat s neurčitými daty a používání přesných popisů by vedlo k idealizování skutečností reálného světa a tedy k odklonu od reality.

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

1.3.1 Modelování neurčitosti

Neurčitost zde chápeme jako vlastnost systému, kdy jej nelze přesně matematicky popsat nebo kdy matematický popis neumožňuje dostatečně přesně predikovat jeho budoucí chování. Obecně se dá říci, že zdrojem neurčitosti je nedostatek informací. Problémem je, že informace, ze kterých při tvorbě modelu vycházíme mohou být nekompletní, vzájemně si odporující, nespolehlivé, vágní nebo jinak nedostačující. Tyto nedostatky v potřebných informacích mají za následek různé druhy neurčitostí. Nejistotou při transformaci systému do matematického modelu rozumíme situaci, kdy nemáme k dispozici všechny potřebné informace nebo kdy některé z informací jsou nespolehlivé.

Pro jednoduchost můžeme neurčitosti ovlivňující model rozdělit na tři spolu související kategorie [31]:

a) **neurčitost v matematickém popisu modelu** je výsledkem nedostatečné znalosti chování modelovaného systému, neúplných exaktních dat, či zjednodušení, které bylo nutné provést pro matematický popis. V literatuře je možné setkat se s pojmy strukturální (konstrukční) chyba, konceptuální chyba, neurčitost v konceptuálním modelu, nebo chyba/neurčitost modelu, které částečně či zcela odpovídají tomuto druhu neurčitosti.

b) **datová neurčitost** neboli neurčitost v datech je způsobena chybami měření, nepřesností analytických metod a omezeným množstvím vzorků při sběru a zacházením s daty. Datovou neurčitost někdy nazýváme odstranitelnou neurčitostí, neboť je možné ji dalším studiem (měřením) minimalizovat. Speciálně pro tento druh neurčitosti používáme v češtině termín **nejistota**.

c) **neurčitost v aplikaci modelu** vyjadřuje neurčitost (chybu), která vznikne použitím modelu. V tomto případě se jedná zejména o řešení matematických rovnic popisujících model pomocí nástrojů ICT.

Analýza neurčitostí vyšetřuje zmíněné neurčitosti ve vstupu a jejich vliv na výstup modelu. Zpravidla se skládá z následujících kroků:

- *charakterizace vstupních neurčitostí* -– odhad neurčitostí ve vstupu a parametrech algoritmu modelu;
- *šíření neurčitostí* -– odhad neurčitosti ve výstupu způsobené neurčitostmi na vstupu modelu;
- *charakterizace neurčitostí modelu* -– charakterizace neurčitostí spojená s jinými strukturami algoritmu modelu a formulacemi modelu;
- *charakterizace neurčitostí v predikcích algoritmu modelu* -– vychází z neurčitostí ve vyhodnocených datech.

Neurčitost má (nejméně) dvě vzájemně komplementární stránky: *vágnost* a *nejistotu*. Vágnost lze modelovat např. pomocí teorie *fuzzy množin*, zatímco nejistotu např. pomocí *teorie pravděpodobnosti* a popř. dalších teorií, jako je teorie možnosti, různé míry věrohodnosti apod. Můžeme tedy říci, že pravděpodobnost nám odpovídá na otázku, zda „něco nastane“, zatímco teorie fuzzy množin nám odpovídá na otázku, „co vlastně nastalo“. Je zřejmé, že při matematickém modelování systému s neurčitostmi se vyskytuje jak nejistota, tak vágnost. Ze studijních účelů však lze obě tyto stránky oddělit a zabývat se pouze jednou z nich.

Podle [24] můžeme říci, že „*Předmětem teorie pravděpodobnosti je studium a modelování nejistoty. Ta nastává tehdy, jestliže se setkáváme s nějakým jevem, který může, avšak nemusí nastat. Nemáme tedy jistotu, že jev opravdu nastane. Základním pojmem v teorii pravděpodobnosti je rozdělení pravděpodobnosti. To charakterizuje způsob nastání jevů vybíraných z nějaké množiny různých jevů, o nichž víme určitě jen to, že jeden z nich nastane. Pravděpodobnost nám pak dává informaci o tom, zda nastání některého z uvažovaných jevů můžeme očekávat s větší jistotou, než nastání jiného jevu.*“

Naproti tomu uvažujme např. *objekty s určitou vlastností* a na otázku, jakou mají „*vlastnost*“ odpovíme, že by ji mohly mít, než, že ji mají či ne. Jde totiž o vymezení vlastnosti a nikoliv toho, zda ji jev má či ne. Základním pojmem je zde fuzzy množina objektů a funkce příslušnosti objektu do ní. To znamená, že prvek patří do množiny s jistou mírou příslušnosti – stupněm příslušnosti. Funkce, která každému prvku universa přiřadí stupeň příslušnosti, se nazývá funkce příslušnosti. Funkce příslušnosti, stejně jako pravděpodobnosti, mohou nabývat hodnot z intervalu $[0, 1]$. To je však jen vnějšková shoda s teorií pravděpodobnosti.

„*Fuzzy logika umožňuje zahrnout nepřesnost a poměrně jednoduchým způsobem pracovat s významy slov přirozeného jazyka. Používá vágně charakterizované expertní znalosti. Tedy pravý opak toho, co se vždy požadovalo – větší přesnost. Narážíme na reálný rozpor, jehož řešení neexistuje. Jde o vztah mezi relevancí a přesností informace. Princip, který L. A. Zadeh nazval principem „inkompatibility“, lze charakterizovat takto: Chceme-li popsat realitu, pak musíme rozhodnout mezi relevancí informace, která bude méně přesná, nebo přesností informace, která však bude méně relevantní. Při zvyšování přesnosti se dostaneme k bodu, kdy přesnost a relevance se stávají vzájemně se vylučujícími charakteristikami.*“ [24]

Příklad: Instrukce k zaparkování auta: „pootoč kola o 19° 25.32’ a popojeď o 368.1256 mm dozadu“ – jednak by se tato informace zdlouhavě a složitě chystala a také by bylo obtížné ji přesně splnit. Stačí říct: „pootoč kola mírně doleva a popojeď o malý kousek dozadu.“ „*K vyjádření relevantní informace je nezbytné použít přirozený jazyk. Je to dosud jediný dokonalý prostředek, který nám umožňuje efektivně pracovat s vágními pojmy.*“ [24]

Ukazuje se, že přesnost matematického modelu je pouze iluze, neboť je principiálně nedosažitelná. Snaha o absolutní přesnost nás vždy dovede ke sporu. Není však třeba litovat,

neboť vágnost, kterou přirozený jazyk umí dokonale využít, je jeho hlavní silou, nikoliv nedostatkem.

Častá námitka, že to, co je řešeno pomocí fuzzy logiky, lze řešit i bez ní, neobstojí. Rozdíl mezi klasickým řešením a řešením pomocí fuzzy logiky je v čase a s tím souvisejících nákladech. Trvalo by to mnohem déle, abychom dosáhli stejného efektu (správnosti). Nalezení matematického popisu může být v praxi velmi obtížné. Často je popis velmi složitý, a následné použití modelu není zrovna snadné. Proto se buď používají přibližné metody nebo se přijímají různá zjednodušení a výsledek pak nemusí být uspokojivý.

Při řešení pomocí fuzzy logiky je navíc větší jistota, že dané řešení (model systému) bude robustnější vzhledem k náhodným poruchám a nepředvídaným situacím, které pochopitelně lze očekávat.

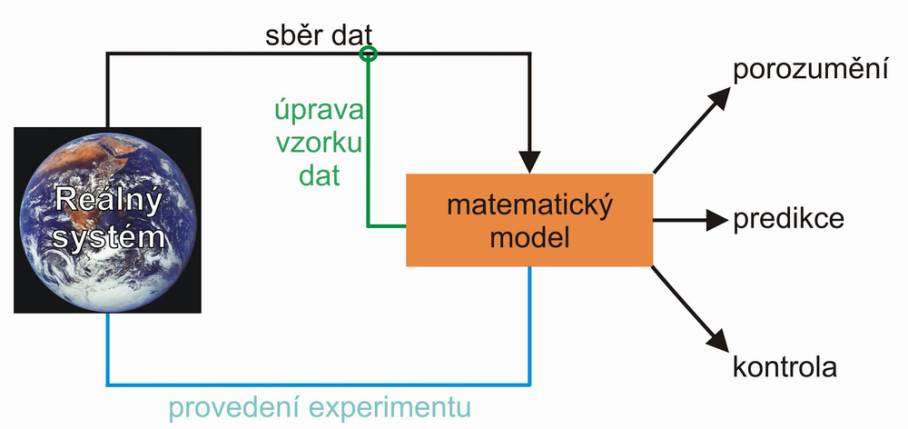
K vyjádření relevantní informace je možné použít přirozený jazyk. S počítačem však není možné komunikovat v přirozeném jazyce, a proto jsou nezbytná jistá zjednodušení. Obvyklý způsob je použití pravidel typu JESTLIŽE-PAK. Tato pravidla jsou základem všech úvah a základem činnosti všech algoritmů.

Příklad: JESTLIŽE sklon svahu je velký a srážky jsou intenzivní, PAK se svah velmi rychle sesune.

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 [24].

2 Metodologie matematického modelování

Na obrázku 2.1 je znázorněn proces zkoumání systému, který začíná monitoringem a sběrem dat o systému, pokračuje jejich vstupem do matematického modelu a na základě analýzy výsledků řešení modelu (porozumění, predikce a kontrola jeho chování), je provedena případná úprava modelu a přizpůsobení sběru dat. Pokud to nevede k uspokojivému řešení, provedou se na zkoumaném systému nové experimenty, případně se modifikuje matematický model.



Obrázek 2.1: Proces zkoumání systému s využitím matematického modelování

2.1 Obecné zásady matematického modelování

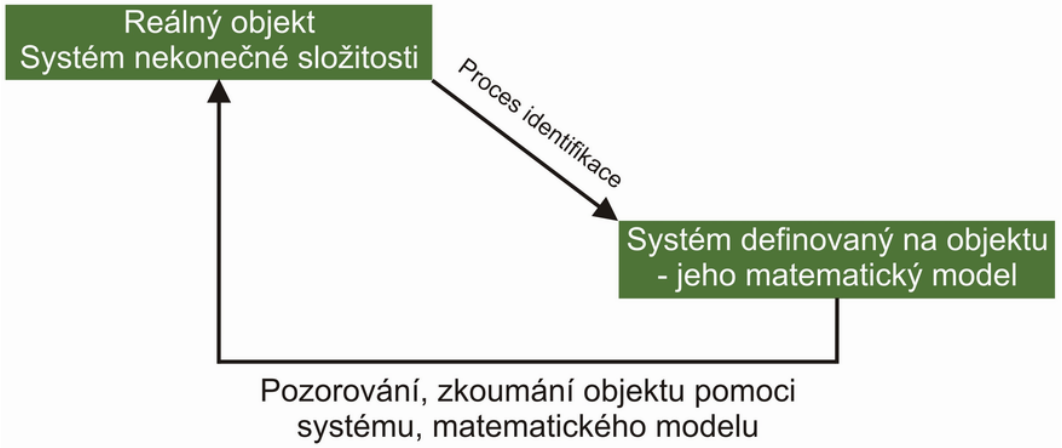
Matematické modelování je odborná a kvalifikovaná činnost vyžadující týmovou spolupráci odborníků z různých oblastí: odborníka z oblasti oboru řešené problematiky, specialistu v oblasti matematiky, specialistu z oblasti informatiky apod. Pro úspěšné matematické modelování musí být splněny následující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

- Znalost metod a prostředků matematické a funkcionální analýzy, algebry a diskrétní matematiky — důležité pro volbu 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 proměnných, parametrů a dat, které vycházejí z konkrétních hodnověrných informací, dat a zdůvodněných odhadů parametrů. Údaje musí být ve vhodné formě pro ověřování modelu. Je vhodné vytvářet speciální informační systémy (např. banky dat), jež uchovávají data.
- Experimentální základna. U složitých matematickým modelů by měla být k dispozici i experimentální základna, kde se řešení modelu ověří v praxi.

Metodologie matematického modelování se vyvíjí jako samostatná odborná specializace, která se v současné době zařazuje do studijních programů Matematického a počítačového modelování na vysokých školách. Systém uvažujeme jako uspořádanou množiny prvků, mezi nimiž působí vzájemné vazby. Obecné zásady, jež je třeba při matematickém modelování systémů respektovat, lze velmi zjednodušeně popsat následujícími kroky, které pak podrobněji popíšeme v další kapitole:

1. Identifikace systému z hlediska matematického modelování sestává z:
   - 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

Identifikace systému je pojem, který se obvykle používaný k popisu matematických nástrojů a algoritmů, které vytváří dynamické modely z naměřených dat. V této publikaci jej budeme uvažovat v obecnějším smyslu.

2. V kroku specifikace modelu systému je nutno určit:

   - Prvky systému: tj. elementární část systému při dané rozlišovací úrovni dále nedělitelná
   - Vazby: vzájemné závislosti mezi prvky systému, tj. kauzální vztahy: příčina-následek, způsoby spojení mezi prvky, souvislosti mezi jevy, informační vazby, matematicky formulované vztahy atd.
   - Strukturu systému: je dána prvky a vazbami mezi nimi.
   - Stavy systému: tj. popsat stavy chování systému a přechodu mezi nimi u dynamických systémů apod.
   - Okolí systému: tj. množinu prvků, které nejsou prvky systému, ale mají k němu významné vazby
   - Organizaci dat v systému: tj. jejich strukturu, způsobu uložení, vstup a výstup dat atd.
   - Verifikaci modelu systému: tj. ověření matematických předpokladů, stability, konzistence a konvergence řešení atd.

   Konkrétní specifikace či formulace matematického modelu je základem celého matematického modelování a záleží do značné míry na schopnostech řešitelského týmu spojit teoretické poznatky s informacemi o konkrétním problému nebo systému, který je předmětem analýzy. V této fázi matematického modelování musí být věnována pozornost i tomu, zda vstupní data skutečně odpovídají proměnným zahrnutým do modelu. V některých případech je totiž vhodnější dát přednost relativně jednoduše specifikovanému modelu před složitým, často zavádějícím modelem.

3. Výpočet řešení modelu sestává z:

   - Volby algoritmu řešení.
   - Výběru variant řešení.

4. Výběr dostatečně vhodných řešení sestává z:

   - Výběru vhodných řešení v rámci algoritmu řešení.
   - Výběru vhodných řešení prováděných specialistou v řešené problematice.
   - Výběru vhodných řešení prováděných skupinou expertů.

5. Experimentování s vybraným řešením sestává z:

   - „What-if“ analýzy, tj. analýzy „Co se stane, když“ používané většinou při numerické simulaci sloužící k odhalení, jak je model citlivý na odhad vstupních parametrů, jež by měly ležet v nějakém vhodném intervalu.
   - „Goal seeking“ problému, tj. „Cílovým hledáním problému“, který souvisí s tím, že v praxi se vytváří matematický model metodou postupných kroků lépe aproximujících 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“ problém.
   - Scénářů, tj. vhodnou volbou parametrů modelu se zkoumají různá řešení (scénáře), z nich se pak vybere nejvhodnější řešení.

6. Výběr optimálního řešení, tj. na základě předem stanovených kritérií se vybere optimální řešení.

7. Implementace, tj. model je nutno implementovat ve vhodném softwarovém prostředí (vývojové prostředí, programovací jazyk, vizualizace řešení apod.). Přitom je nutno:

   - Sledovat postup implementace modelu.
   - Připravit experimentální data s cílem ověřit implementovaný model a provést jejich validaci.
   - Analyzovat počítačové výstupy řešení modelu a mít zpětnou vazbu na parametry modelu při volbě scénářů.
   - Na základě získaných výsledků provést úpravy modelu a jeho novou implementaci.

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. Jde o permanentní interaktivní proces s četnými zpětnými vazbami, který se několikrát opakuje [11]. Nejprve je nutno vyjasnit si cíle, které chceme dosáhnout. Ty určí směr řešení ve dvou bodech:

- Na jaké úrovni podrobnosti chceme zkoumat objekt.
- 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.

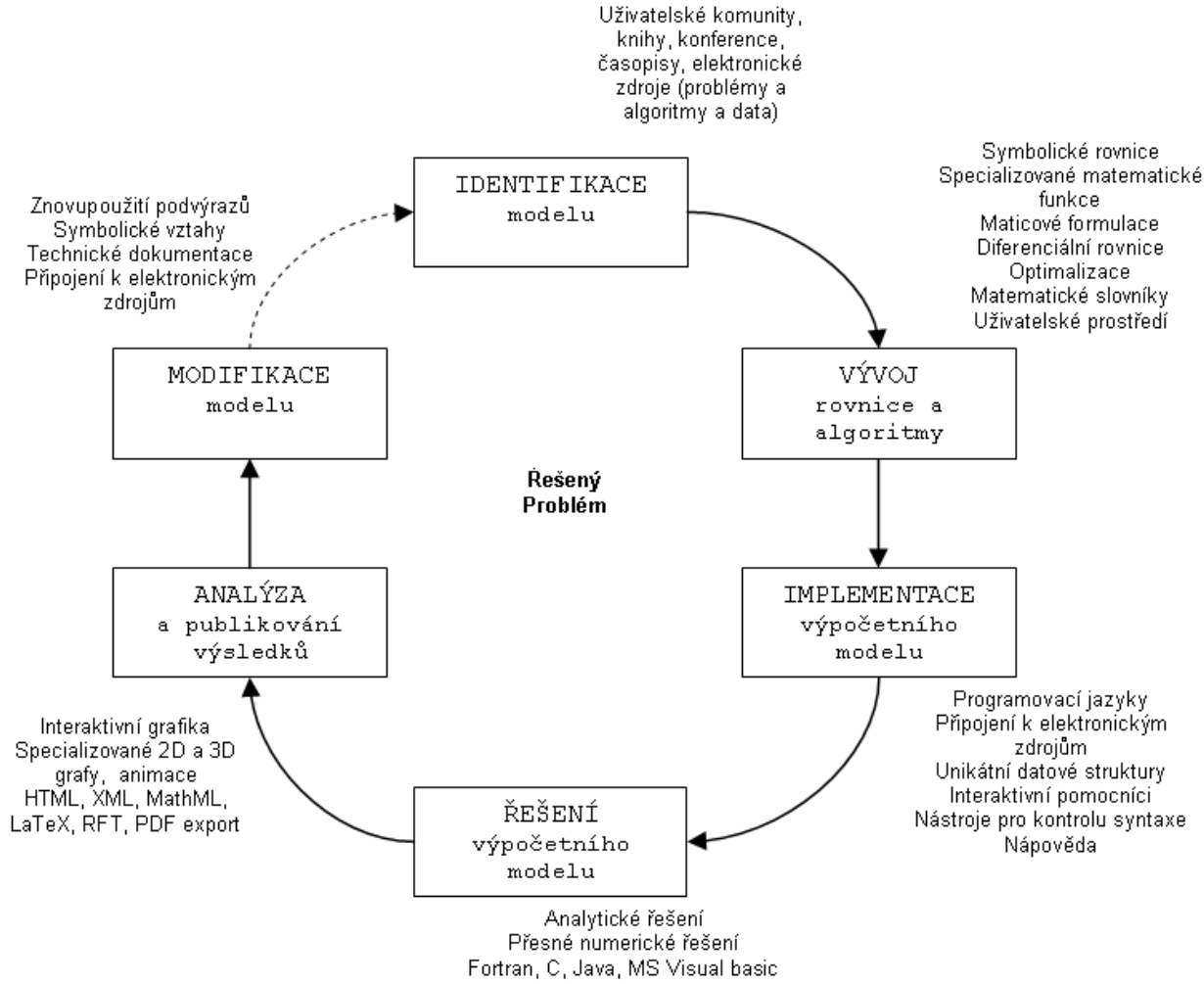
2.2.1 Identifikace modelu

Identifikace (stanovení) jednotlivých prvků matematického modelu s využitím odborné literatury (knihy, časopisy, Internet), spoluprací 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 jeho celkového 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, stavy a funkce. Dále provádíme zjednodušení (simplifikaci) řešeného 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. 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 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 dostatečně precizní, mohou okamžitě vést přímo k matematickým rovnicím popisujícím modelovaný systém.



Obrázek 2.3: Matematické modelování s využitím ICT

2.2.2 Sestavení modelu

Sestavení modelu (vývoj matematických rovnic a formulí) včetně matematické analýzy (korektnost, konzistence, stabilita a konvergence řešení) 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 (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 také dobrou znalost věcné problematiky. Každý model musí vycházet z konkrétní hypotézy odvozené z objektivní reality (skutečnosti).

Výběr matematických rovnic

Poté, 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 odborné literatury a Internetu

Může se stát, že někdo další publikoval matematické rovnice týkající se problému, o nějž se zajímáme. To poskytuje dobrý výchozí bod, ale je nezbytné postupovat s těmito informacemi obezřetně. Problémy, s nimiž se můžeme setkat, jsou 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.

2.2.3 Implementace modelu

Implementace modelu s využitím ICT (naprogramování v příslušném programovacím jazyce, jeho odladění a verifikace, analýza jeho výpočetní složitosti, 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:

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

b) 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 Monte Carlo, simulace na počítači apod.). Při numerickém řešení musíme uvažovat jeho numerickou stabilitu, 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.

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.

3 Populační modely

Teorii matematického modelování popsanou v úvodních kapitolách nyní aplikujeme na konkrétním případě tzv. populačních modelů. Budeme vytvářet modely populací živých organismů, které žijí v daném čase a prostředí. Naším cílem bude vytvořit model odpovídající na otázku, kolik jedinců bude mít populace v daném čase $t > 0$, jestliže známe tento počet na počátku (v čase $t = 0$).

Modely růstu populace patří k nejrozšířenějším a nejznámějším, tudíž bychom mohli nyní převzít nějaký již vytvořený model z odborné literatury (např. [15]) nebo ze zdrojů na Internetu (např. [19], [27]). Předpokládejme však, že žádný takový model ještě vytvořen nebyl a popíšeme podrobně jednotlivé kroky jeho tvorby.

3.1 Růst populace živých organismů

3.1.1 Identifikace a sestavení modelu

Nejprve budeme modelovat růst jedné populace. Cíl modelu jsme již stanovili. Nyní musíme specifikovat jednotlivé prvky modelu a definovat předpoklady, za nichž bude model platit. Modelujeme růst jedné populace P v čase t, která sestává z počtu N jedinců a na počátku (v čase $t = 0$) měla $N0$ jedinců. Předpokládejme, že změna velikosti N populace P v čase je způsobena pouze plozením nových jedinců a umíráním jiných. Přitom počet nově narozených, respektive zemřelých, jedinců bude přímo úměrný velikosti populace. Pro jednoduchost budeme uvažovat, že na populaci P nepůsobí žádné další vlivy.

Hledejme řešení modelu, tj. velikost N populace P v daném čase t. Čas t budeme uvažovat jednak jako diskrétní veličinu nabývající celočíselných hodnot (mohou představovat například roky), nebo jako spojitou veličinu.

Na základě vyslovených předpokladů jsme schopni sestavit rovnici modelu. Označme:

$N(t)$ ... funkci představující počet jednotlivců populace (velikost populace) v čase t
a ... *koeficient porodnosti* populace (podíl nově narozených jedinců ke všem jedincům za jednotku času)
b ... *koeficient úmrtnosti* populace (podíl zemřelých jedinců ke všem jedincům za jednotku času)
h ... kladné reálné číslo představující časový interval

Vzhledem k smysluplnosti modelu předpokládáme, že všechny výše uvedené proměnné mohou nabývat pouze reálných nezáporných hodnot, navíc $b \leq 1$ (nemůže zemřít více jedinců, než kolik jich je v populaci). Počet jedinců $N(t)$ vypočítaný pro daný čas t nemusí být celé číslo, při interpretaci výsledků jej proto budeme zaokrouhlovat. Navíc předpokládáme, že modelujeme růst populace od času $t = 0$, v němž známe hodnotu velikosti populace $N(0) = N0$.

Počet nově narozených jedinců za časový interval $[t, t+h]$ je přímo úměrný počtu jedinců $N(t)$ populace v čase t a délce h tohoto intervalu, tj:

$$a \cdot N(t) \cdot h,$$

kde koeficientem přímé úměrnosti a je koeficient porodnosti. Analogické tvrzení platí pro počet zemřelých jedinců za daný časový interval, tj:

$$b \cdot N(t) \cdot h,$$

kde koeficientem přímé úměrnosti b je koeficient úmrtnosti.

Nyní již můžeme formulovat rovnice modelu.

Diskrétní případ

Čas t v tomto případě nabývá pouze celočíselných hodnot, takže h bude rovněž celé kladné číslo. Pro jednoduchost zvolme $h = 1$. Rovnice modelu má pak tvar[16]:

$$N(t+1) \;=\; N(t) \;+\; a \cdot N(t) \cdot 1 \;-\; b \cdot N(t) \cdot 1 \;=\; (1 + a - b) \cdot N(t). \tag{3.1}$$

s počáteční podmínkou

$$N(0) = N0. \tag{3.2}$$

Spojitý případ

Nechť h je nyní kladné reálné číslo. Potom má rovnice modelu tvar:

$$N(t+h) \;=\; N(t) \;+\; a \cdot N(t) \cdot h \;-\; b \cdot N(t) \cdot h. \tag{3.3}$$

Jelikož uvažujeme spojitý čas, tak můžeme předpokládat, že časový interval $[t, t+h]$ je nekonečně malý. Proto upravíme rovnici (3.3) na tvar:

$$\frac{N(t+h) - N(t)}{h} = a \cdot N(t) - b \cdot N(t) = (a - b) \cdot N(t).$$

A odtud limitním přechodem $(h \to 0)$ dostaneme:

$$\lim_{h \to 0} \frac{N(t+h) - N(t)}{h} = (a - b) \cdot N(t). \tag{3.4}$$

Výraz na levé straně rovnice (3.4) je derivace funkce $N(t)$ podle proměnné t, takže výsledná rovnice modelu má tvar[17,18]:

[16] Tomuto typu rovnic říkáme rovnice rekurentní (případně diferenční) [17].

[17] Tomuto typu rovnic říkáme rovnice diferenciální [16].

[18] Derivaci funkce podle času budeme dále značit apostrofem, tj. $\frac{d}{dt}N(t) = N'(t)$.

$$\frac{d}{dt}N(t) = (a - b) \cdot N(t) \tag{3.5}$$

s počáteční podmínkou

$$N(0) = N0. \tag{3.6}$$

Z biologického hlediska lze limitně přejít k diferenciální rovnici (3.5) pouze, pokud jsou splněny následující podmínky [9]:

- *kvantovací podmínka*: populace je tak velká, že není třeba počítat s jedinci,
- *vzorkovací podmínka*: všichni jedinci v populaci jsou identičtí (populace je homogenní z hlediska jedinců v produkčním věku).

3.1.2 Implementace modelu a jeho řešení

Řešení výše uvedeného matematického modelu (rovnic modelu) můžeme vypočítat sami na základě svých znalostí, většinou však využíváme vhodných informačních technologií. My budeme používat systém počítačové algebry Maple[19] (více o práci v systému v kapitole 5 Maple). Uveďme si tedy, jak implementujeme vytvořený model a jak získáme řešení.

Diskrétní případ

V systému Maple zapíšeme rovnici (3.1) s počáteční podmínkou (3.2). Klikneme na ni pravým tlačítkem myši a zvolíme z nabídky příkaz na řešení rekurentních rovnic: **Solve Recurrence > N(t)**, viz obrázek 3.1. Získáme tak řešení na obrázku 3.2.

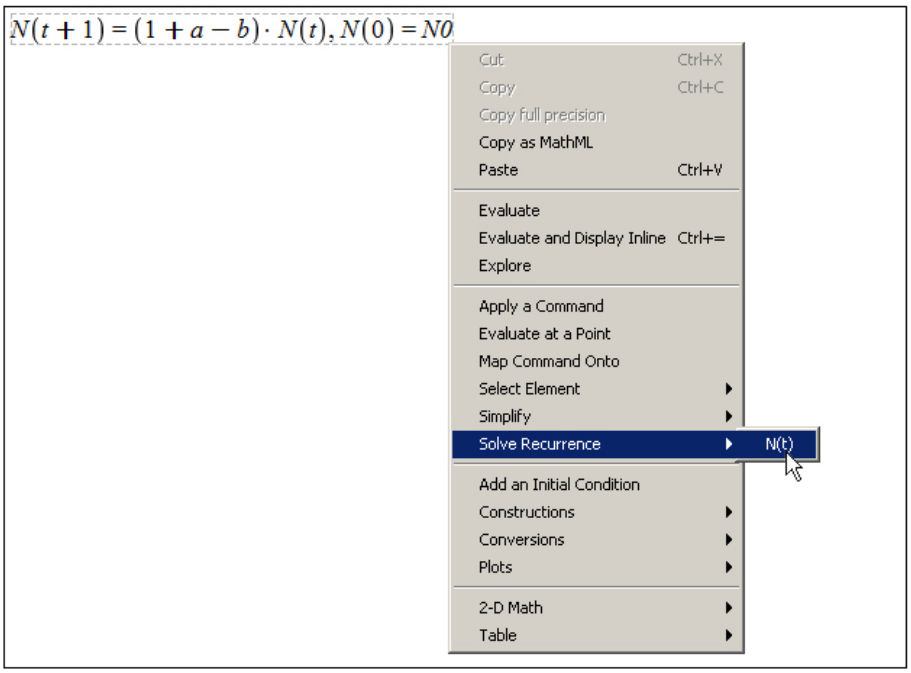
Spojitý případ

Zcela analogicky zapíšeme v systému Maple rovnici (3.5) s počáteční podmínkou (3.6). Poté na ni klikneme pravým tlačítkem myši a z nabídky vybereme příkaz na řešení diferenciálních rovnic: **Solve DE > N(t)**, viz obrázek 3.3. Získáme řešení na obrázku 3.4.

3.1.3 Analýza řešení

Nyní můžeme přistoupit k analýze výsledků řešení modelu. Využijeme k tomu grafické možnosti systému Maple.

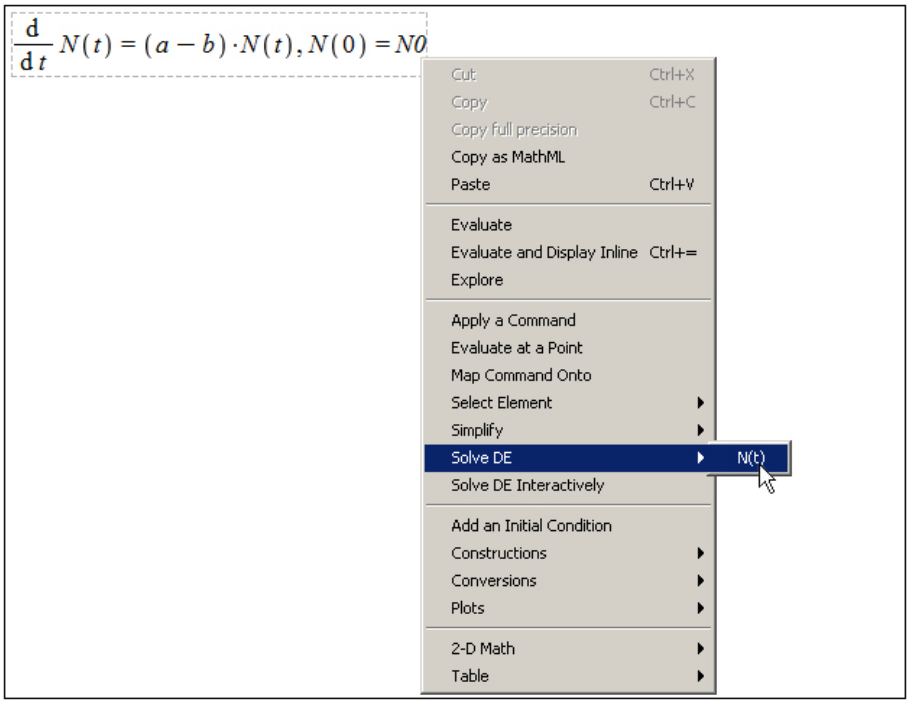
[19]Konkrétně se jedná o verzi Maple 14. U starších verzí systému je nutné některé kroky řešení provést jinak (zpravidla použít příkazy jazyka Maple namísto tlačítka myši) k získání stejného výsledku. Kvůli jednotnému zápisu oddělovače desetinných míst u čísel budeme v textu používat vždy desetinnou tečku.



Obrázek 3.1: Řešení rovnice (3.1) v systému Maple

$$N(t+1)=(1+a-b)\cdot N(t), N(0)=N0 \xrightarrow{\text{solve recurrence}} N0\,(1+a-b)^{t}$$

Obrázek 3.2: Řešení rovnice (3.1) s počáteční podmínkou (3.2)



Obrázek 3.3: Řešení rovnice (3.5) v systému Maple

$$\frac{\mathrm{d}}{\mathrm{d}\,t} N(t) = (a-b)\cdot N(t), N(0) = N0 \xrightarrow{\text{solve DE}} N(t) = N0\,\mathrm{e}^{(a-b)\,t}$$

Obrázek 3.4: Řešení rovnice (3.5) s počáteční podmínkou (3.6)

Diskrétní případ

Abychom mohli vykreslit řešení v diskrétním případě, je nutné přiřadit parametrům modelu $N0,\ a,\ b$ konkrétní numerické hodnoty. Uvažujme proto populaci zajíce polního. Z literatury [23] zjistíme, že samice zajíce polního vrhá 3–4 krát do roka 1–7 mláďat a že se zajíc polní dožívá 10–12 let.

Na základě těchto údajů odhadneme hodnoty koeficientů porodnosti a úmrtnosti. Předpokládáme přitom, že samic je v populaci stejný počet jako samců a plození nových jedinců je průměrné. Samice zajíce tedy 3.5krát v roce vrhne 4 mláďata. To je celkem 14 nově narozených jedinců za rok. Jelikož je v populaci samic polovina, poměr nově narozených vzhledem k celkovému počtu jedinců populace bude „pouze“ 7. Jak zjistíme dále, tato hodnota je velmi vysoká. Proto pro větší přehlednost přejdeme k délce kroku rovnou jednomu měsíci. Koeficient porodnosti tak bude dvanáctina z původně vypočítané hodnoty, tj. $a = \frac{7}{12}$.

Podobně odvodíme koeficient úmrtnosti. Zajíc polní se dožívá průměrně 11 let, za rok tedy zemře $\frac{1}{11}$ jeho populace. Opět přejdeme k měsíčnímu kroku a získáme $b = \frac{1}{11\cdot 12}$. Nechť počáteční velikost populace $N0$ je 100 jedinců. Pro časový krok rovný jednomu měsíci celkem máme:

$$N0 = 100, \quad a = \frac{7}{12}, \quad b = \frac{1}{11\cdot 12}.$$

Řešení vykreslíme následujícím postupem. Vypočítáme hodnoty řešení pro časové body, které nás zajímají (např. $t = 1, 2, ..., 10$), a příkazem **plot** pro vykreslování je zobrazíme. Příkazu **plot** dále nastavíme několik nepovinných parametrů (*symbol, symbolsize, labels, font, labelfont*) pro lepší vzhled a popis souřadných os. Ukázku zdrojového kódu s výsledky poskytuje obrázek 3.5.

Spojitý případ

Ve spojitém případě lze s výhodou využít pomocníka pro vykreslování **Plot Builder** a zkoumat závislost řešení na hodnotách parametrů modelu. V systému Maple zapíšeme získané řešení na obrázku 3.4, klikneme na něj pravým tlačítkem myši a z nabídky vybereme zmíněného pomocníka přes **Plots > Plot Builder**. Na obrazovce se nám objeví okénko, v němž nastavíme, co nás zajímá. Zcela nahoře vybereme z rolovacího seznamu možnost **Interactive Plot with 3 parameters**. Následně zvolíme v níže umístěném rámečku **2-D plot** (tato možnost by měla být zvolena standardně, tudíž není nutné nic měnit).

Nakonec nastavíme rozpětí hodnot parametrům modelu tak, jak chceme. Nejprve přiřadíme ose x proměnnou t a ostatní pole vyplníme zbylými parametry s jejich požadovanými rozsahy. Ná základě dříve uvedených údajů zvolme: $t \in [0, 10],\ N0 \in [50, 500],\ a \in [0, 1.17],\ b \in [0.007, 0.0083]$, přičemž připustíme, že koeficient porodnosti může být i nulový, abychom mohli pozorovat vývoj řešení i v případě, kdy je koeficient porodnosti nižší než koeficient úmrtnosti. Horní mez koeficientu porodnosti odpovídá situaci, kdy samice zajíce

Rovnice modelu a její řešení:

$N(t+1) = (1+a-b)\cdot N(t), N(0) = N0 \xrightarrow{\text{solve recurrence}} N0\,(1+a-b)^t$

Nastavení parametrů modelu: $N0 := 100 : \; a := \frac{7}{12} : \; b := \frac{1}{11\cdot 12} :$

Vytvoření funkce počtu jedinců: $N(t) := N0\cdot\,(1+a-b)^t$

$t \to N0\,(1+a-b)^t$

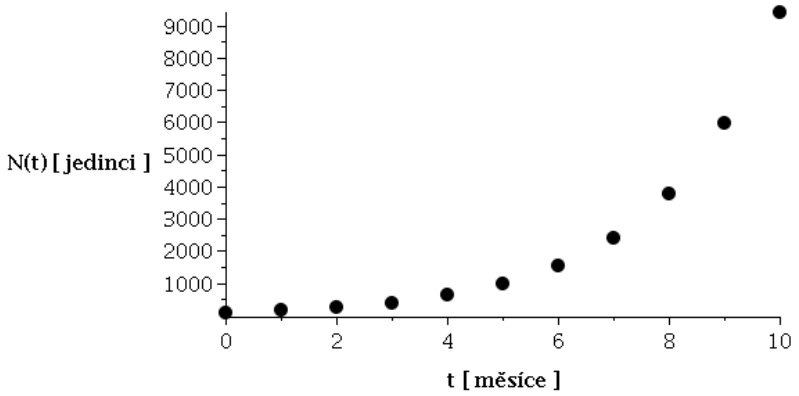
Vytvoření posloupnosti hodnot řešení po měsících:

$posloupnost := seq([t, evalf(N(t))], t = 0..10)$

[0, 100.], [1, 157.5757576], [2, 248.3011938], [3, 391.2624871], [4, 616.5348282], [5, 971.5094263], [6, 1530.863338], [7, 2412.269503], [8, 3801.151944], [9, 5989.693972], [10, 9438.305653]

Vykreslení řešení:

$plot([posloupnost], style = point, symbol = solidcircle, symbolsize = 20, labels = ["t [měsíce]", "N(t) [jedinci]"], font = [Times, roman, 12], labelfont = [Times, bold, 12])$



Obrázek 3.5: Řešení diskrétního případu modelu růstu populace.

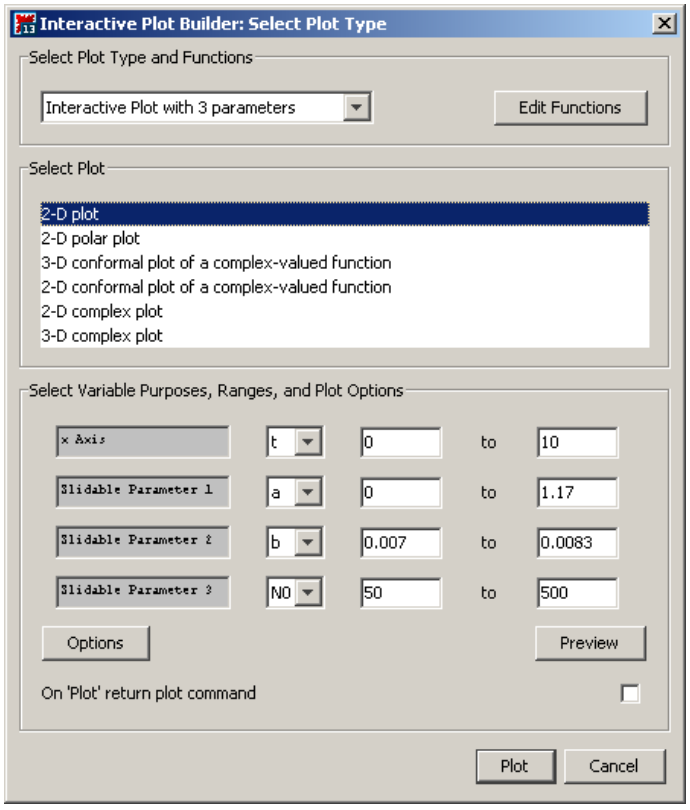
vrhne 4krát do roka 7 mláďat. Analogickou úvahou jako dříve dospějeme k „měsíční“ hodnotě $\frac{14}{12} \doteq 1.17$. Meze koeficientu úmrtnosti odpovídají situacím, kdy se zajíc polní dožije 10 a 12 let. Podobu vyplněného okénka podle předchozího postupu ilustruje obrázek 3.6.

Následným kliknutím na tlačítko **Plot** přejdeme na okénko se zobrazením řešení pro nastavení parametrů $a, b, N0$. Toto nastavení je možné měnit posuvníky po pravé straně okénka. Při změně libovolného parametru se automaticky přizpůsobí vykreslení řešení nové hodnotě parametru. Okénko s vykreslením řešení poskytuje obrázek 3.7.

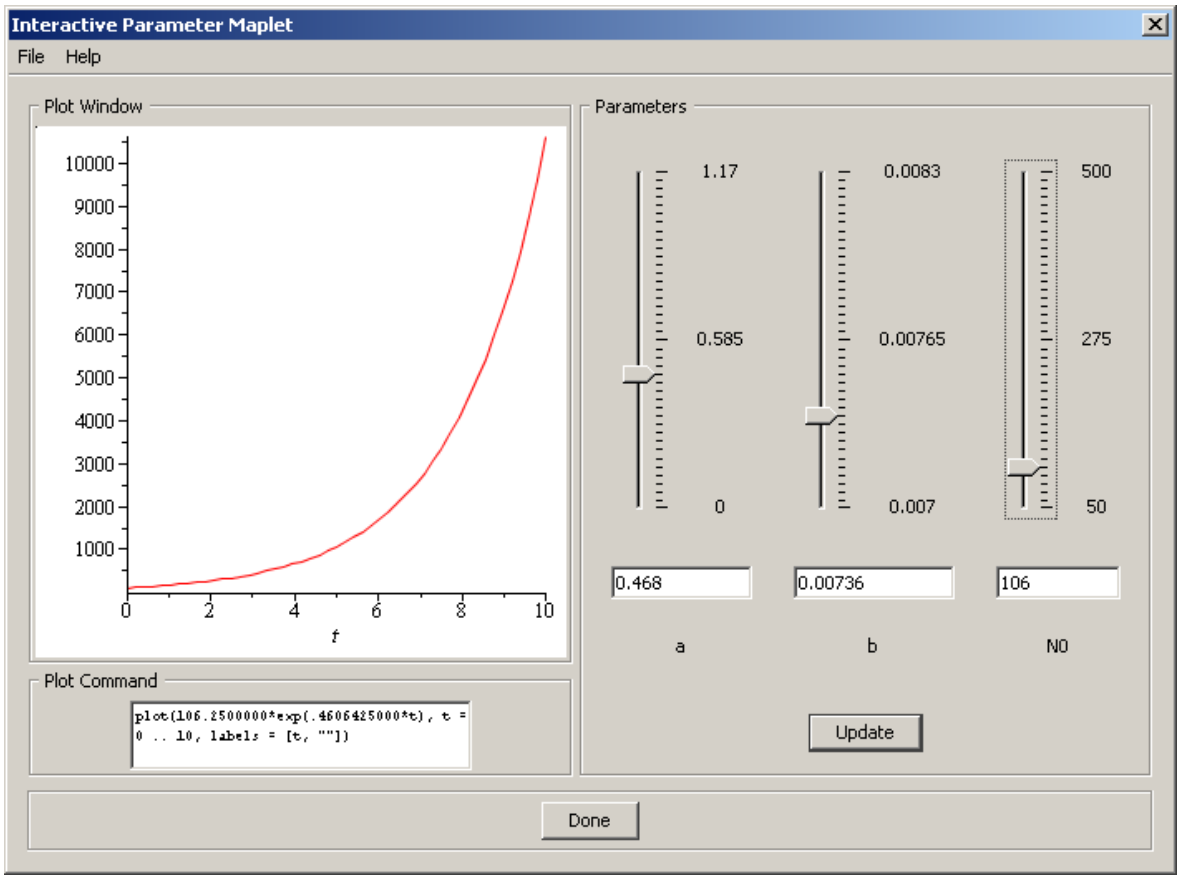
Po stisknutí tlačítka **Done** umístíme graf z vykreslovacího okénka do zápisníku systému Maple a současně tím vykreslovací okénko zavřeme.

Z předešlého zkoumání závislosti řešení na parametrech modelu můžeme dospět k následujícím závěrům. Pokud bude koeficient porodnosti vyšší než koeficient úmrtnosti, velikost populace bude exponenciálně růst. Pokud budou hodnoty koeficientů totožné, populace bude mít stále stejnou velikost. A jestliže bude koeficient úmrtnosti vyšší než koeficient porodnosti, populace bude vymírat. Počáteční velikost populace nemá na zmíněné chování řešení žádný vliv (pokud je vyšší než nula).

V diskrétním případě je situace prakticky totožná. Řešení se chová kvalitativně stejně, „malé“ rozdíly můžeme pozorovat v hodnotách velikosti populace v jednotlivých časových bodech.



Obrázek 3.6: Nastavení Plot Builderu pro vykreslení spojitého řešení.



Obrázek 3.7: Vykreslení spojitého řešení.

3.1.4 Modifikace modelu

Nyní porovnáme výsledky modelu s pozorováním a kriticky zhodnotíme model. Je zřejmé, že problém růstu populace je složitější, než jak jsme si jej namodelovali. Provedeme proto vylepšení modelu, které jej více přiblíží realitě.

Velikost populace se nemůže exponenciálně zvyšovat do nekonečna. Prostor, v němž populace žije, je omezený, podobně jako množství živin, které má k dispozici. Doplňme proto předpoklad modelu, že úmrtnost se bude zvyšovat se zvětšující se populací. Nejjednodušší způsob závislosti je lineární závislost. Koeficient úmrtnosti tedy nebudeme již chápat jako konstantní číslo, ale jako rostoucí lineární funkci.

Koeficient úmrtnosti: $b + c \cdot N(t)$, kde b, c jsou reálná nezáporná čísla.

Podobně jako dříve získáme rovnice modelu.

Diskrétní případ: $N(t+1) = (1 + a - b) \cdot N(t) - c \cdot N(t)^2,\ N(0) = N0$

Spojitý případ: $N'(t) = (a - b) \cdot N(t) - c \cdot N(t)^2,\ N(0) = N0$

V diskrétním případě se nám nepodaří získat analytické řešení modelu jako dříve (obrázek 3.2), z rekurentního předpisu však můžeme určit velikost populace v libovolném čase t. Zaměřme se nejprve na spojitý případ, v němž je analýza řešení jednodušší.

Řešení rovnice modelu v systému Maple obdržíme zapsáním rovnice, kliknutím pravého tlačítka myši a zvolením příkazu pro řešení diferenciálních rovnic **Solve DE**. Výsledek tohoto postupu je znázorněn na obrázku 3.8.

$$\frac{\mathrm{d}}{\mathrm{d}t} N(t) = (a-b)\cdot N(t) - c\cdot (N(t))^2, N(0) = N0 \xrightarrow{\text{solve DE}} N(t) = -\frac{N0\,(a-b)}{-N0\,c + \mathrm{e}^{-(a-b)\,t} N0\,c - \mathrm{e}^{-(a-b)\,t} a + \mathrm{e}^{-(a-b)\,t} b}$$

Obrázek 3.8: Řešení modifikované rovnice modelu – spojitý případ

Analýzu řešení provedeme znovu za pomoci **Plot Builderu**. V systému Maple zapíšeme před chvílí získané řešení (můžeme zkopírovat) a po kliknutí pravým tlačítkem myši vybereme z nabídky **Plots** > **Plot Builder**. Naprosto totožným způsobem jako dříve nastavíme rozsahy parametrů a zkoumáme závislost řešení na jejich hodnotách ovládáním posuvníků.

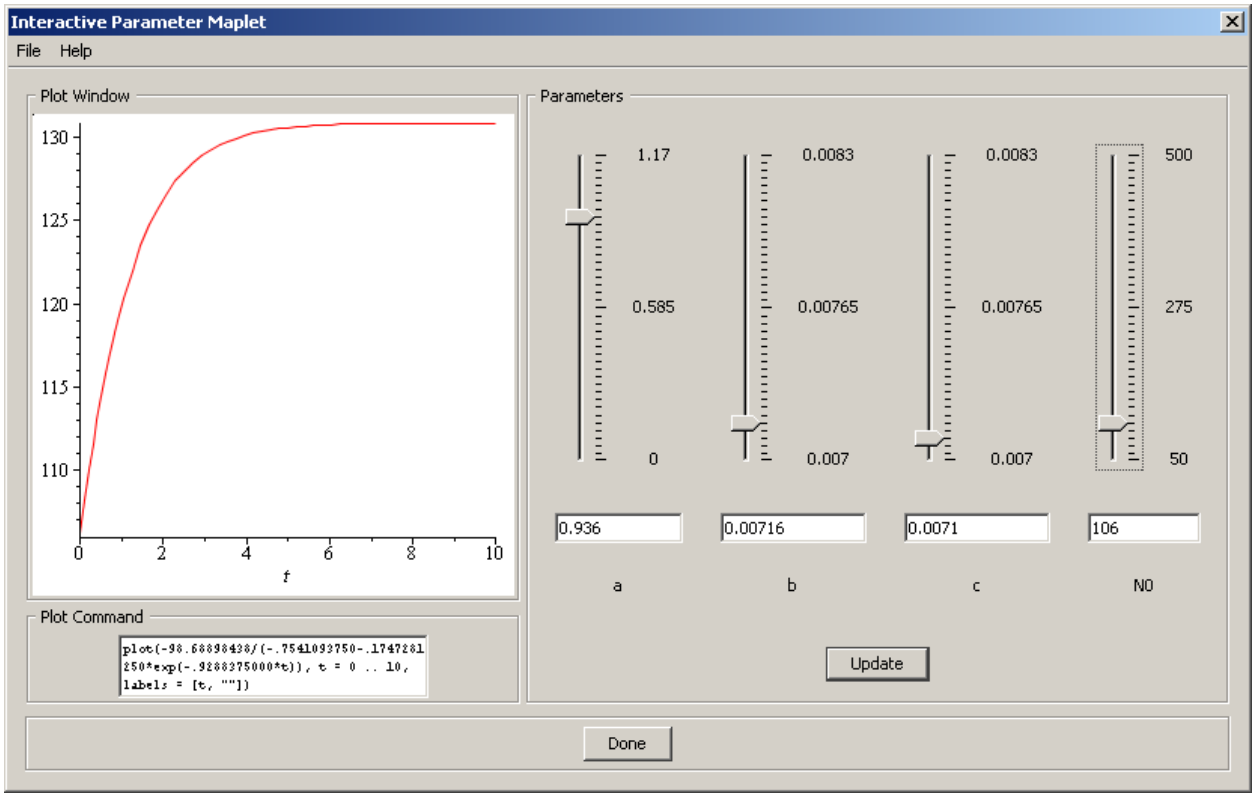
Rozsahy všech dříve použitých parametrů necháme stejné, hodnota parametru c závisí na prostředí, v němž populace žije. Nechť v našem případě nabývá tento parametr stejného rozsahu jako parametr b. Obrázek 3.9 ilustruje vykreslovací okénko spolu s nastavením parametrů pomocí posuvníků.

Nastavováním různých hodnot parametrům pomocí posuvníků v **Plot Builderu** zjistíme, že velikost populace se vždy (po „dostatečně“ dlouhém čase) ustálí na nějaké hodnotě, při níž je populace v dynamické rovnováze s prostředím (neroste ani nevymírá). Tuto hodnotu můžeme určit z rovnice modelu položením levé strany rovné nule, tj. derivace (změna) velikosti populace v čase je nulová:

$$0 = (a - b) \cdot N(t) - c \cdot N(t)^2. \tag{3.7}$$

Rovnice (3.7) má jediné nenulové řešení, a to:

$$N(t) = \frac{a - b}{c}.$$



Obrázek 3.9: Vykreslení řešení modifikované rovnice modelu – spojitý případ

Pokud $a \geq b$, velikost populace se ustálí na hodnotě $\frac{a-b}{c}$ (tj. $\lim_{t\to\infty} N(t) = \frac{a-b}{c}$). Pokud $a < b$, populace vymře (stejně jako v předcházejícím modelu). Zmíněné chování platí pro libovolnou nenulovou počáteční velikost populace.

Právě odvozenou ustálenou hodnotu velikosti populace $\left(\frac{a-b}{c}\right)$ nazýváme *úživností* (resp. *kapacitou*) *prostředí* a značíme písmenem K. Označme dále

$$r = a - b.$$

Tento parametr, tj. rozdíl koeficientů porodnosti a úmrtnosti, se nazývá *vnitřní koeficient růstu populace*.

Při tomto označení můžeme spojitý případ rovnice původního modelu (3.5) přepsat na tvar:

$$N'(t) = r \cdot N(t) \tag{3.8}$$

a spojitý případ rovnice modifikovaného modelu na tvar:

$$N'(t) = r \cdot \left(1 - \frac{N(t)}{K}\right) \cdot N(t). \tag{3.9}$$

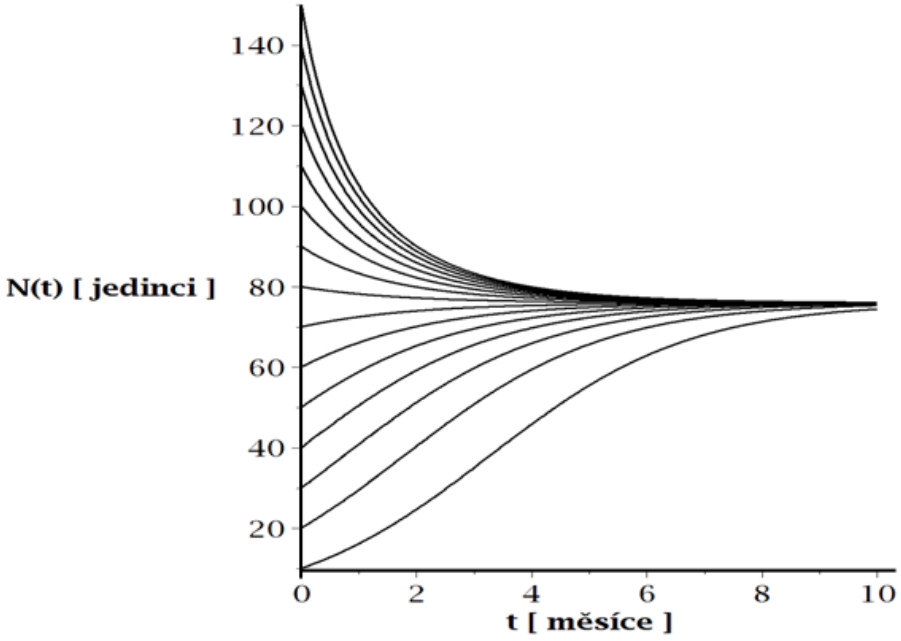
Chování řešení v případě modifikovaného modelu závisí na počáteční velikosti populace. Velikost populace v čase klesá, pokud $N(0) > K$, roste, pokud $N(0) < K$, případně zůstává konstantní. Závislost řešení na počáteční velikosti populace dokumentuje obrázek 3.10, v němž $K = 76$ a $N0 \in \{15, 30, ..., 150\}$. Pomocí **for**-cyklu je vygenerováno 10 řešení pro jednotlivá $N0$, která jsou vykreslena v jednom grafu příkazem **display** z balíku **plots**.

Nastavení parametrů: $a := \frac{7}{12} : b := \frac{1}{11 \cdot 12} : c := \frac{1}{11 \cdot 12} :$

Definice funkce řešení: $N(t) := -\frac{N0\,(a-b)}{-N0\,c + \mathrm{e}^{-(a-b)\,t} N0\,c - \mathrm{e}^{-(a-b)\,t} a + \mathrm{e}^{-(a-b)\,t} b} :$

Cyklus vytvoření grafů pro jednotlivé hodnoty *N0*:
for *i* **from** 1 **to** 15 **do**
N0 := *i*·10 :
graf[*i*] := *plot*(*N*(*t*), *t* = 0 ..10, *thickness* = 3, *labels* = ["t [měsíce]", "N(t) [jedinci]"], *font* = [*Times*, *roman*, 12], *labelfont* = [*Times*, *bold*, 12], *numpoints* = 1500) :
end do:

Vykreslení grafů:
plots[*display*](*seq*(*graf*[*i*], *i* = 1 ..15))



Obrázek 3.10: Vykreslení závislosti řešení na počáteční velikosti populace

Vraťme se nyní k diskrétnímu případu modifikovaného modelu, který je možné po zavedení parametrů K a r přepsat do tvaru:

$$N(t+1) = N(t) + r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right), \; N(0) = N0.$$

Analýza diskrétních případů populačních modelů bývá složitější. Nejinak je tomu v tomto případě. Pro jednoduchost uvažujme situaci, kdy je nenulová počáteční velikost populace $N0$ nižší než úživnost prostředí K pro tuto populaci. V závislosti na velikosti vnitřního koeficientu růstu r můžeme pozorovat čtyři (kvalitativně) různá řešení (viz [27]): velikost populace monotónně roste k hodnotě úživnosti prostředí (obrázek 3.11), přibližuje se k ní s tlumenými oscilacemi (obrázek 3.12), kolísá kolem této hodnoty pravidelně (obrázek 3.13), nebo kolem této hodnoty kolísá nepravidelně, chaoticky (obrázek 3.14). Dříve použitým hodnotám parametrů r a K (u spojitého případu) odpovídá první situace (obrázek 3.11).

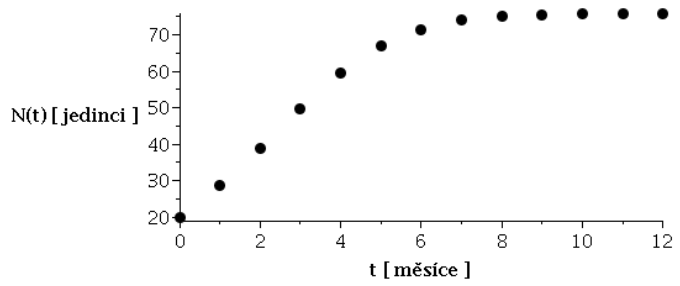
Rovnice modelu: $N(t+1) = N(t) + r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right)$:

Nastavení parametrů modelu: N0 := 20 : r := $\frac{7}{12} - \frac{1}{11 \cdot 12}$: K := 76 :

Výpočet velikosti populace v časových bodech *t* = 1, ..., 12:
Velikost[1] := *N0* :
for *i* **from** 2 **to** 13 **do**
Velikost[*i*] := *evalf*$\left(Velikost[\,i-1\,] + r \cdot Velikost[\,i-1\,] \cdot \left(1 - \frac{Velikost[\,i-1\,]}{K}\right)\right)$
end do:

Vytvoření posloupnosti hodnot řešení:
posloupnost := *seq*([*t*, *Velikost*[*t* + 1]], *t* = 0 ..12) :

Vykreslení řešení:
plot([*posloupnost*], *style* = *point*, *symbol* = *solidcircle*, *symbolsize* = 20, *labels* = ["t [měsice]",
"N(t) [jedinci]"], *font* = [*Times*, *roman*, 12], *labelfont* = [*Times*, *bold*, 12])



Obrázek 3.11: Řešení diskrétního případu modifikovaného modelu růstu populace pro $0 < r < 1$.

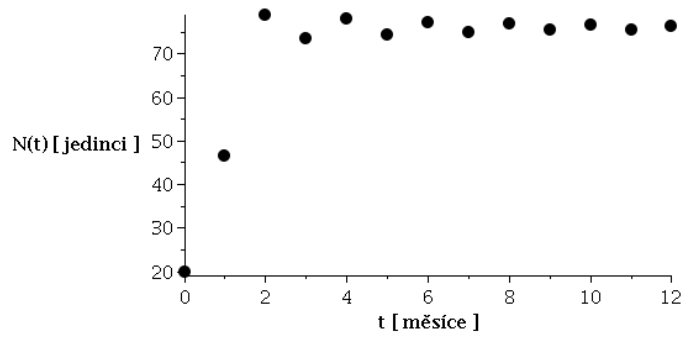
Rovnice modelu: $N(t+1) = N(t) + r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right)$:

Nastavení parametrů modelu: N0 := 20 : r := 1.8 : K := 76 :

Výpočet velikosti populace v časových bodech *t* = 1, ... , 12:
Velikost[1] := *N0* :
for *i* **from** 2 **to** 13 **do**
Velikost[*i*] := *evalf*$\left(Velikost[\,i-1\,] + r \cdot Velikost[\,i-1\,] \cdot \left(1 - \frac{Velikost[\,i-1\,]}{K}\right)\right)$
end do:

Vytvoření posloupnosti hodnot řešení:
posloupnost := *seq*([*t*, *Velikost*[*t* + 1]], *t* = 0 ..12) :

Vykreslení řešení:
plot([*posloupnost*], *style* = *point*, *symbol* = *solidcircle*, *symbolsize* = 20, *labels* = ["t [měsice]",
"N(t) [jedinci]"], *font* = [*Times*, *roman*, 12], *labelfont* = [*Times*, *bold*, 12])



Obrázek 3.12: Řešení diskrétního případu modifikovaného modelu růstu populace pro $1 < r < 2$.

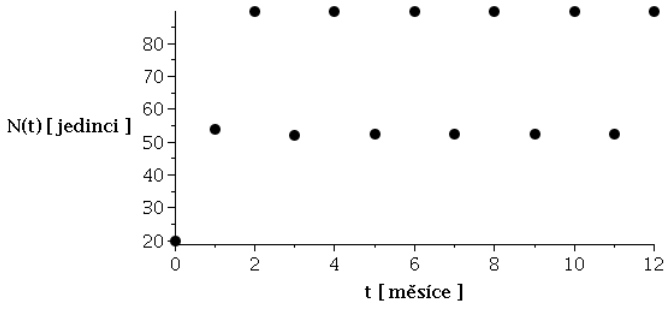
Rovnice modelu: $N(t+1) = N(t) + r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right)$:

Nastavení parametrů modelu: N0 := 20 : r := 2.3 : K := 76 :

Výpočet velikosti populace v časových bodech t = 1, ... , 12:
$Velikost[1] := N0$:
for i **from** 2 **to** 13 **do**
$Velikost[i] := evalf\left(Velikost[i-1] + r \cdot Velikost[i-1] \cdot \left(1 - \frac{Velikost[i-1]}{K}\right)\right)$
end do:

Vytvoření posloupnosti hodnot řešení:
$posloupnost := seq([t, Velikost[t+1]], t = 0..12)$:

Vykreslení řešení:
$plot([posloupnost], style = point, symbol = solidcircle, symbolsize = 20, labels = ["t [měsice]", "N(t) [jedinci]"], font = [Times, roman, 12], labelfont = [Times, bold, 12])$



Obrázek 3.13: Řešení diskrétního případu modifikovaného modelu růstu populace pro $2 < r < 2.8$.

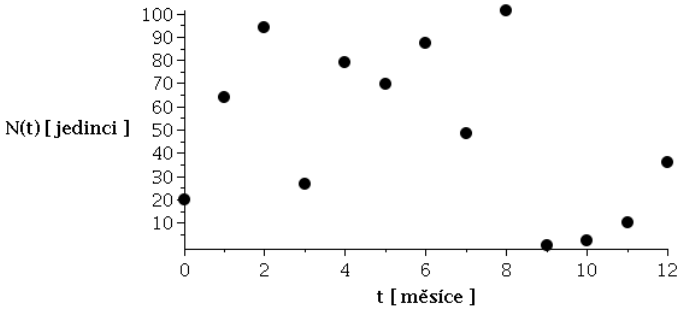
Rovnice modelu: $N(t+1) = N(t) + r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right)$:

Nastavení parametrů modelu: N0 := 20 : r := 3 : K := 76 :

Výpočet velikosti populace v časových bodech t = 1, ..., 12:
$Velikost[1] := N0$:
for i **from** 2 **to** 13 **do**
$Velikost[i] := evalf\left(Velikost[i-1] + r \cdot Velikost[i-1] \cdot \left(1 - \frac{Velikost[i-1]}{K}\right)\right)$
end do:

Vytvoření posloupnosti hodnot řešení:
$posloupnost := seq([t, Velikost[t+1]], t = 0..12)$:

Vykreslení řešení:
$plot([posloupnost], style = point, symbol = solidcircle, symbolsize = 20, labels = ["t [měsice]", "N(t) [jedinci]"], font = [Times, roman, 12], labelfont = [Times, bold, 12])$



Obrázek 3.14: Řešení diskrétního případu modifikovaného modelu růstu populace pro $r > 2.8$.

3.2 Populace pod tlakem nespecializovaného predátora

3.2.1 Identifikace a sestavení modelu

V předešlém příkladu jsme modelovali růst populace, která není ovlivněna svým okolím. Nyní přejdeme k situaci, kdy je v prostředí, v němž se uvažovaná populace vyvíjí, také nespecializovaný predátor. Nespecializovaný predátor je takový, který není závislý na kořisti z uvažované populace, má i alternativní zdroje obživy. Velikost jeho populace tedy můžeme považovat za konstantní a do modelu ji nemusíme zahrnovat.

První předpoklad při tvorbě modelu bude přirozený: predátoři uloví takové množství kořisti, které je úměrné době lovu, tj. množství ulovené kořisti za časový interval délky h je rovno $p \cdot h$. Parametr p se nazývá *intenzita predace* a vyjadřuje predační tlak vyvíjený na uvažovnou populaci, přesněji řečeno: množství kořisti, které predátoři uloví za jednotku času. Intenzita predace závisí na velikosti N populace kořisti, tj. $p = p(N)$. Abychom tuto závislost vyjádřili, přijmeme další dva přirozené předpoklady:

- pokud není uvažovaná populace přítomná, predátoři nic neuloví a živí se alternativní potravou,
- pokud je uvažovaná populace veliká (větší než predátoři dokáží sníst), loví predátoři jen omezené množství jedinců, které představuje jakousi *hladinu nasycení.*

Tyto předpoklady zapíšeme ve tvaru rovností:

$$p(0) = 0, \qquad p(N) = S \quad \text{pro} \quad N > N_{\text{krit}},$$

kde parametr S představuje hladinu nasycení, N_{krit} kritickou hodnotu velikosti populace (je-li velikost uvažované populace větší než kritická, predátoři jsou nasycení). Zvolme za p nejjednodušší funkci, která splňuje uvedené předpoklady, tedy funkci lineární na intervalu $[0, N_{\text{krit}}]$, jinak konstantní:

$$p(N) = \begin{cases} \dfrac{S}{N_{\text{krit}}} \cdot N & \ldots \quad N \leq N_{\text{krit}}, \\ \\ S & \ldots \quad N > N_{\text{krit}}. \end{cases} \tag{3.10}$$

U předešlého modelu jsme v diskrétním případě dospěli k rovnici

$$N(t+1) \;=\; N(t) \;+\; r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right), \quad N(0) = N0 \tag{3.11}$$

a ve spojitém případě k rovnici

$$N'(t) \;=\; r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right), \quad N(0) = N0. \tag{3.12}$$

Obě tyto rovnice lze za použití časového kroku h vyjádřit ve tvaru

$$N(t+h) \;=\; N(t) \;+\; r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) \cdot h, \quad N(0) = N0. \tag{3.13}$$

Výraz

$$r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) \cdot h$$

lze interpretovat jako přirozený přírůstek velikosti populace za časový interval délky h. Rovnice (3.11) je rovnicí (3.13) pro $h = 1$, rovnice (3.12) je mezním případem rovnice (3.13) pro $h \to 0$.

Nyní vyjádříme změnu velikosti populace za časový interval délky h jako přirozený přírůstek populace zmenšený o množství ulovených jedinců. Rovnici (3.13) tedy dále modifikujeme a dostaneme model růstu populace pod tlakem nespecializovaného predátora ve tvaru

$$N(t+h) \;=\; N(t) \;+\; r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) \cdot h \;-\; p\big(N(t)\big) \cdot h, \quad N(0) = N0, \tag{3.14}$$

kde funkce p je dána rovností (3.10). Pro $h = 1$ dostaneme model diskrétní:

$$N(t+1) \;=\; N(t) \;+\; r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) \;-\; p\big(N(t)\big), \quad N(0) = N0, \tag{3.15}$$

limitním přechodem $h \to 0$ dostaneme model spojitý:

$$N'(t) \;=\; r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) \;-\; p\big(N(t)\big), \quad N(0) = N0. \tag{3.16}$$

3.2.2 Implementace modelu a jeho řešení

S rostoucí složitostí modelu vzrůstá i náročnost výpočtu řešení. Podobně jako v předchozím modifikovaném modelu nejsme schopni určit řešení diskrétního případu (3.15), z rekurentního předpisu však můžeme (při znalosti numerických hodnot parametrů modelu) určit hodnotu velikosti populace v libovolném čase t. Ve spojitém případě je možné rovnici (3.16) rozložit na dva případy (podle (3.10)) a v Maple již známým postupem (**Solve DE > N(t)**) nalézt řešení pro každý případ zvlášť. Tento postup je však možný jedině tehdy, kdy je velikost populace stále „pod hladinou“ N_{krit}, nebo stále nad ní (připouštíme i rovnost).

V obecném případě je nutné řešit rovnici (3.16) za pomoci numerických metod, k čemuž musíme znát numerické hodnoty všech parametrů modelu. Můžeme opět uvažovat populaci zajíce polního, na jehož území nyní žije i liška obecná jakožto nespecializovaný predátor. Musíme však zjistit hodnoty použitých parametrů. Od této chvíle se pro zbytek textu odkloníme od konkrétních hodnot naměřených v praxi a pro jednoduchost a názornost budeme volit „vymyšlené“ (avšak realistické) nastavení parametrů. V grafech proto dále nebudeme uvádět jednotky (které se mohou lišit pro různé modelované populace), neboť nám jde o kvalitativní chování populací. V tuto chvíli zvolme například následující nastavení:

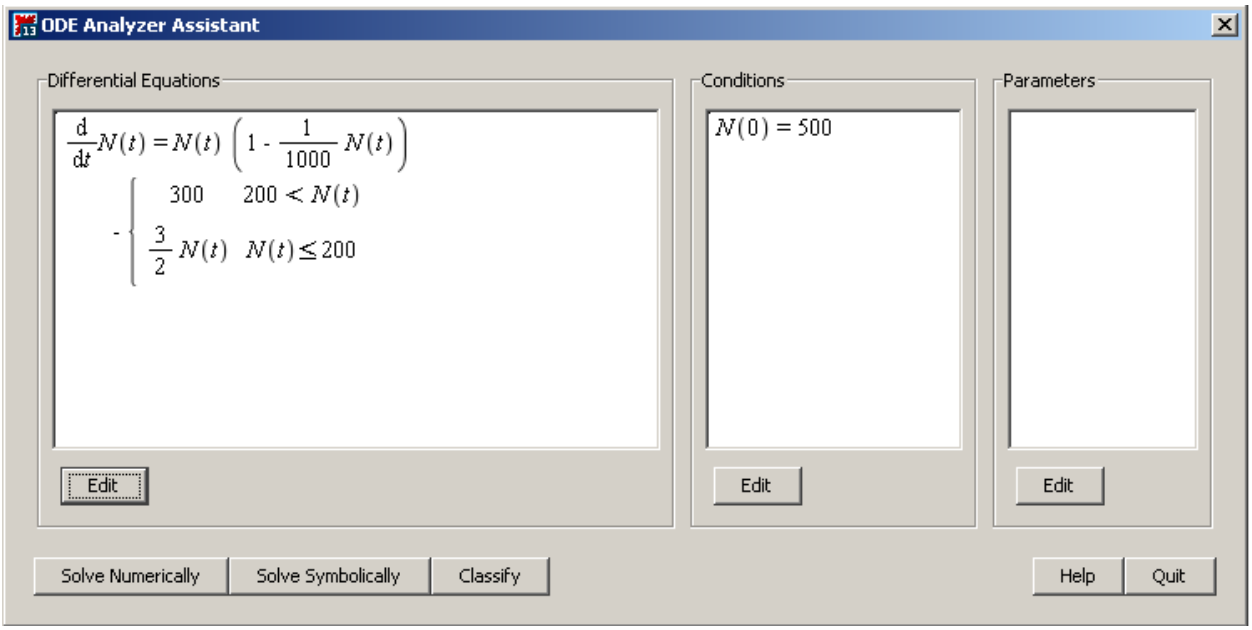
$$S = 300,\ r = 1,\ K = 1000,\ N_{\text{krit}} = 200,\ N0 = 500.$$

V systému Maple nastavíme proměnným příslušné hodnoty a zapíšeme rovnici ve tvaru znázorněném na obrázku 3.15. K zápisu dvou možných případů pro funkci $p(N(t))$ přitom

Nastavení parametrů: $S := 300 : \quad r := 1 : \quad K := 1000 : \quad Nkrit := 200 : \quad N0 := 500 :$

Rovnice modelu: $$\frac{\mathrm{d}}{\mathrm{d}t}N(t) = r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit} \cdot N(t) & N(t) \le Nkrit \end{cases}, \quad N(0) = N0$$

Obrázek 3.15: Zápis rovnice (3.16).



Obrázek 3.16: ODE Analyzer Assistant.

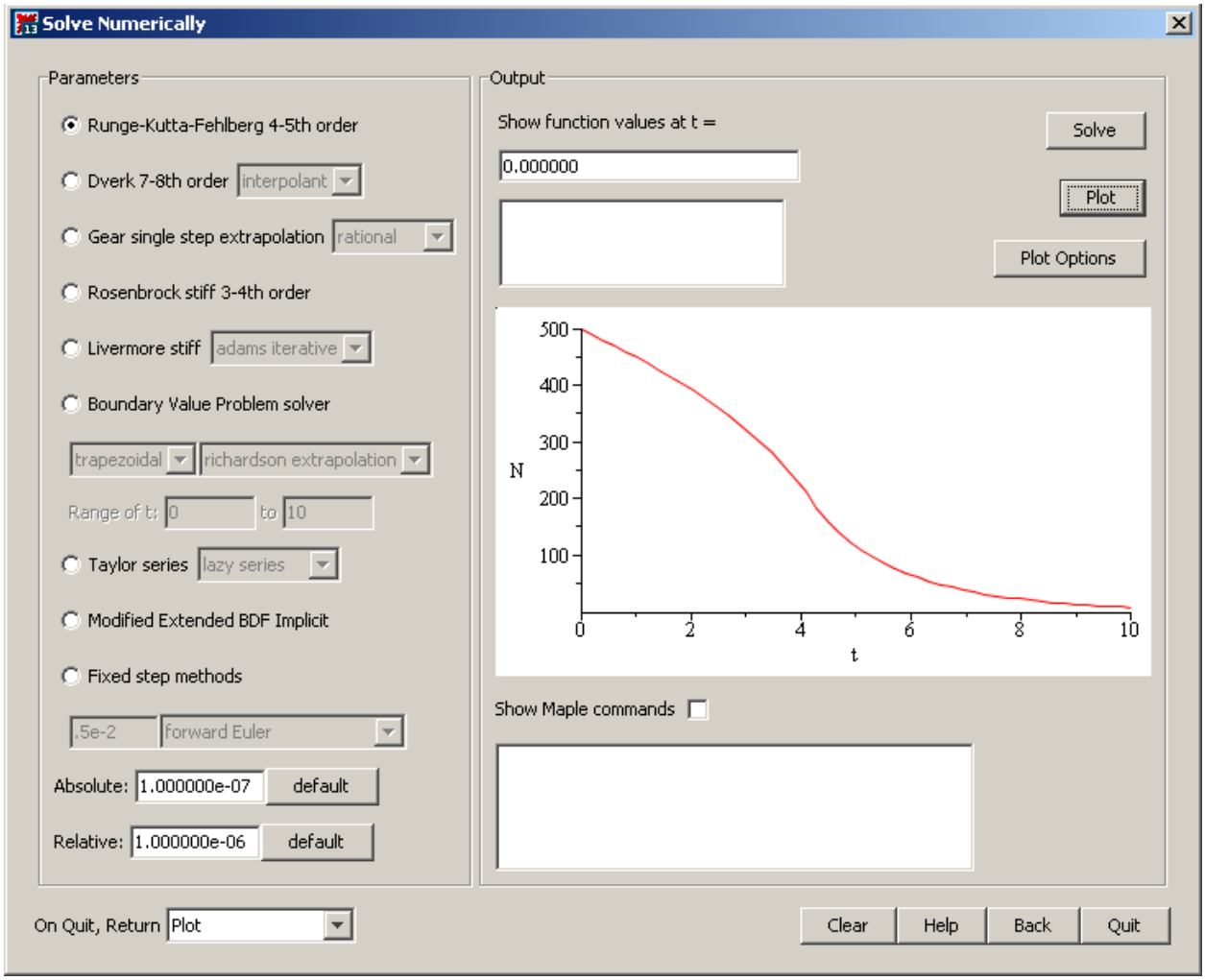
využijeme symbolu tvaru otevřené složené závorky v paletě **Exression** reprezentující výraz definovaný po částech.

Po kliknutí pravým tlačítkem myši na rovnici a zvolením **Solve DE Interactively** se nám zobrazí okénko **ODE Analyzer Assistant** s upraveným tvarem rovnice a počáteční podmínkou (obrázek 3.16). V tomto okénku je možné rovnici dále upravovat, měnit počáteční podmínky, zavádět a rušit parametry modelu atd. Máme již vše připravené, a tak klikneme na tlačítko **Solve Numerically** pro numerický výpočet řešení.

Přejdeme tak do dalšího okénka, jež po levé straně nabízí numerické metody, které můžeme použít k nalezení řešení, a nastavení absolutní a relativní chyby výpočtu. Využijeme standardního nastavení (tj. nebudeme nic měnit) a klikneme na tlačítko **Plot** pro vykreslení řešení. Okénko s vykresleným řešením poskytuje obrázek 3.17. Po stisknutí tlačítka **Quit** se vrátíme zpět do zápisníku Maple, do nějž bude vložen graf řešení.

3.2.3 Analýza řešení

Víme, že v případě, kdy modelovanou populaci neohrožuje žádný predátor, se velikost populace ustálí na hodnotě K představující úživnost prostředí. Podívejme se, na jaké hodnotě se ustálí velikost populace nyní.



Obrázek 3.17: Numerické řešení rovnice (3.16).

Obě rovnice (3.15), (3.16) vyjadřují změnu velikosti populace v čase. Ustálená hodnota řešení je přitom taková, která se v čase nemění, tj. v diskrétním případě pro ni platí:

$$N(t+1) - N(t) = 0 \tag{3.17}$$

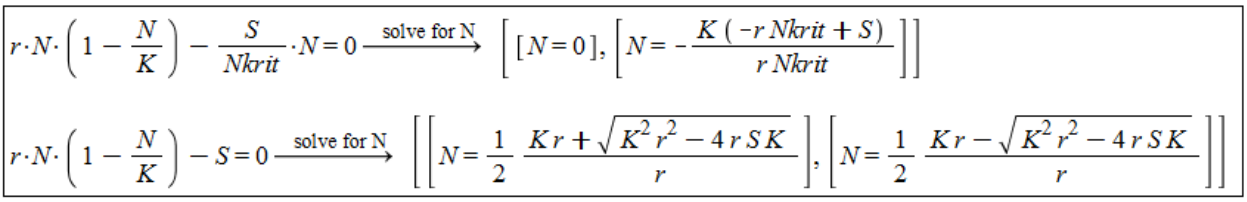
a ve spojitém:

$$N'(t) = 0. \tag{3.18}$$

Obojí vede na rovnici

$$r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) \; - \; p\big(N(t)\big) = 0. \tag{3.19}$$

Rovnici (3.19) vyřešíme v systému Maple pro oba případy (viz (3.10)). Jelikož hledáme ustálenou (konstantní) hodnotu funkce $N(t)$, použijeme v zápisu rovnice symbol N (namísto $N(t)$). Na rovnici klikneme pravým tlačítkem myši a zvolíme **Solve** > **Solve for Variable** > **N**. Výsledek poskytuje obrázek 3.18.



Obrázek 3.18: Řešení rovnice (3.19).

Získaná nenulová řešení ještě upravíme a označíme:

$$N_1 = K \cdot \left(1 - \frac{S}{r \cdot N_{\text{krit}}}\right),$$

$$N_2 = \frac{1}{2} \cdot K \cdot \left(1 - \sqrt{1 - \frac{4 \cdot S}{r \cdot K}}\right), \quad N_3 = \frac{1}{2} \cdot K \cdot \left(1 + \sqrt{1 - \frac{4 \cdot S}{r \cdot K}}\right).$$

Stejně jako u předchozího modifikovaného modelu je analýza diskrétní varianty modelu složitější. Opět se proto nejprve pustíme do analýzy spojitého případu, o diskrétní variantě se zmíníme později.

V závislosti na hodnotě počáteční velikosti populace a vztahu mezi parametry modelu se velikost populace v čase ustálí na jedné ze tří právě nalezených hodnot $N1$, $N2$, $N3$, případně populace vymře (její velikost se ustálí na hodnotě 0). Celkem je možné rozlišit 5 různých kvalitativních situací. Pro každou z nich následuje obrázek řešení s vytvořujícím kódem v systému Maple.

Pro nalezení řešení modelu je použit příkaz **dsolve** s parametrem **numeric**, jenž zajistí numerické řešení příslušné diferenciální rovnice. Grafy řešení jsou vytvořeny příkazem **odeplot** patřícího do balíku **plots** a ukládány do pole (s názvem `graf`), které je následně vykresleno najednou příkazem **display** z balíku **plots**. Příkazu **odeplot** jsou (podobně jako dříve) nastaveny některé nepovinné parametry pro lepší vzhled grafu a popis souřadných os.[20]

1. $S \geq \frac{1}{4} \cdot r \cdot K$

   a) $N_{\text{krit}} \leq \dfrac{S}{r}$

   Predátor vyhubí uvažovanou populaci při jakékoliv počáteční hodnotě $N0 \geq 0$ – obrázek 3.19 pro nastavení parametrů:

   $$S = 300,\ r = 1,\ K = 1000,\ N_{\text{krit}} = 200,\ N0 \in \{50, 100, ..., 1000\}$$

[20] Parametrem **numpoints** specifikujeme, v kolika bodech má být vypočítáno řešení, které je následně vykresleno. Standardně je tento parametr nastaven na 200. Více o vykreslování grafů v kapitole 5 Maple.

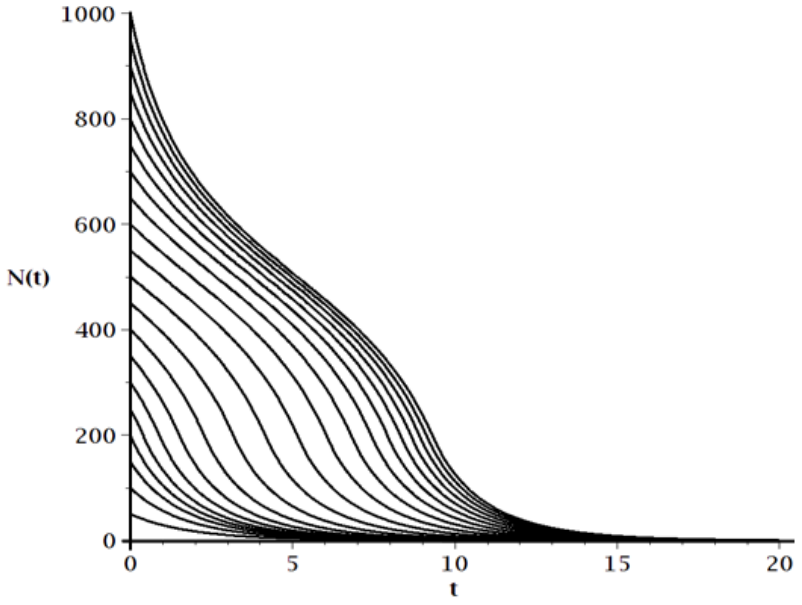
Nastavení parametrů: $S := 300:\ r := 1:\ K := 1000:\ Nkrit := 200:$

Cyklus vytvoření grafů pro jednotlivé hodnoty *N0*:
for i **from** 1 **to** 20 **do**
$N0 := i \cdot 50:$

$$res[i] := dsolve\left(\left[\frac{d}{dt}N(t) = r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit} \cdot N(t) & N(t) \le Nkrit \end{cases}, N(0) = N0\right], numeric\right):$$

$graf[i] := plots[odeplot](res[i], t = 0..20, thickness = 3, labels = ["t [měsice]", "N(t) [jedinci]"], font = [Times, roman, 12], labelfont = [Times, bold, 12], numpoints = 1500):$
end do:

Vykreslení grafů:
$plots[display](seq(graf[i], i = 1..20))$



Obrázek 3.19: Řešení modelu růstu populace pod tlakem nespecializovaného predátora v případě 1. a)

b) $N_{\text{krit}} > \dfrac{S}{r}$:

Velikost populace se ustálí na hodnotě N_1 při jakékoliv počáteční hodnotě $N0 > 0$, tj. predátor zmenší ustálenou velikost populace (z hodnoty K) – obrázek 3.20 pro nastavení parametrů:

$$S = 300,\ r = 1,\ K = 1000,\ N_{\text{krit}} = 400,\ N0 \in \{50, 100, ..., 1000\}$$
$$\Rightarrow N_1 = 250$$

Nastavení parametrů: $S := 300:\ r := 1:\ K := 1000:\ Nkrit := 400:$

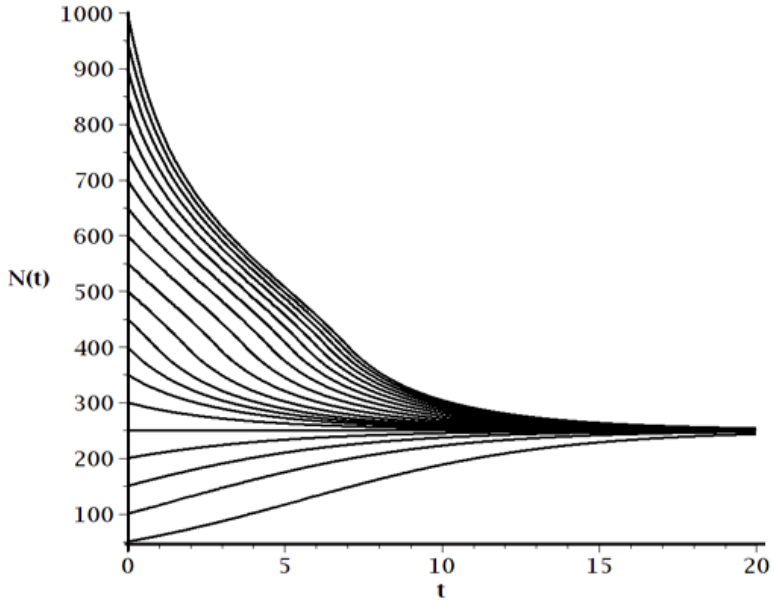
Cyklus vytvoření grafů pro jednotlivé hodnoty *N0*:
for i **from** 1 **to** 20 **do**
$N0 := i \cdot 50:$

$$res[i] := dsolve\left(\left[\frac{d}{dt}N(t) = r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit} \cdot N(t) & N(t) \le Nkrit \end{cases}, N(0) = N0\right], numeric\right):$$

$$graf[i] := plots[odeplot](res[i], t = 0..20, thickness = 3, labels = ["t", "N(t)"], font = [Times, roman, 12],$$
$$labelfont = [Times, bold, 12], numpoints = 1500):$$

end do:

Vykreslení grafů:
$plots[display](seq(graf[i], i = 1..20))$



Obrázek 3.20: Řešení modelu růstu populace pod tlakem nespecializovaného predátora v případě 1. b)

2. $S < \frac{1}{4} \cdot r \cdot K$

   a) $N_{\text{krit}} > N2$:

   Velikost populace se ustálí na hodnotě N_3 při jakékoliv počáteční hodnotě $N0 > 0$, tj. predátor zmenší ustálenou velikost populace (z hodnoty K) – obrázek 3.21 pro nastavení parametrů:

$$S = 200,\ r = 1,\ K = 1000,\ N_{\text{krit}} = 400,\ N0 \in \{50, 100, ..., 1000\}$$
$$\Rightarrow N_1 = 500, \quad N_2 \doteq 276, \quad N_3 \doteq 724$$

Nastavení parametrů: $S := 200:\ r := 1:\ K := 1000:\ Nkrit := 400:$

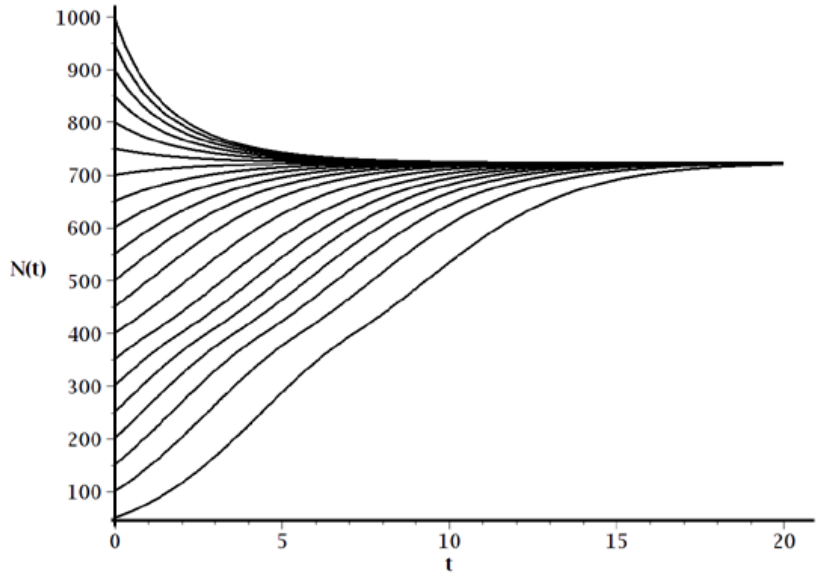
Cyklus vytvoření grafů pro jednotlivé hodnoty *N0*:
for i **from** 1 **to** 20 **do**
$N0 := i \cdot 50:$

$$res[i] := dsolve\left(\left[\frac{d}{dt}N(t) = r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit} \cdot N(t) & N(t) \le Nkrit \end{cases}, N(0) = N0\right], numeric\right):$$

$$graf[i] := plots[odeplot](res[i], t = 0..20, thickness = 3, labels = ["t", "N(t)"], font = [Times, roman, 12],$$
$$labelfont = [Times, bold, 12], numpoints = 5000):$$

end do:

Vykreslení grafů:
$plots[display](seq(graf[i], i = 1..20))$



Obrázek 3.21: Řešení modelu růstu populace pod tlakem nespecializovaného predátora v případě 2. a)

b) $\frac{S}{r} < N_{\text{krit}} \le N2$:

Pokud $N0 > N_2$, velikost populace se ustálí na hodnotě N_3. Pokud $N0 < N_2$, ustálí se na hodnotě N_1. Pro $N0 = N_2$ zůstane velikost populace stejná. Predátor tedy opět zmenší ustálenou velikost populace; nová ustálená velikost populace však závisí na její počáteční velikosti – obrázky 3.22 a 3.23 pro nastavení parametrů:

$$S = 200,\ r = 1,\ K = 1000,\ N_{\text{krit}} = 400,\ N0 \in \{50, 100, ..., 1000\}$$
$$\Rightarrow N1 = 200, \quad N2 \doteq 276, \quad N3 \doteq 724$$

Tuto možnost lze také interpretovat takto: velikost dynamicky stabilizované populace závisí na historii. Pokud populace invadovala do prostředí obsazeného populací predátora, je stabilizovaná populace malá. Naopak, pokud do prostředí se

stabilizovanou populací invadovala populace predátora, je stabilizovaná populace velká.

Nastavení parametrů: $S := 200:\ \ r := 1:\ \ K := 1000:\ \ Nkrit := 250:$

Cyklus vytvoření grafů pro jednotlivé hodnoty *N0*:
for i **from** 1 **to** 20 **do**

$$N0 := i \cdot 50:$$

$$res[i] := dsolve\left(\left[\frac{d}{dt}N(t) = r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit} \cdot N(t) & N(t) \le Nkrit \end{cases}, N(0) = N0\right], numeric\right):$$

$$graf[i] := plots[odeplot](res[i], t = 0..20, thickness = 3, labels = ["t", "N(t)"], font = [Times, roman, 12], labelfont = [Times, bold, 12], numpoints = 1500):$$

end do:

Vytvoření grafu pro hodnotu $N0 = N_2$.

$$N0 := \frac{K}{2} \cdot \left(1 - \sqrt{1 - \frac{4 \cdot S}{r \cdot K}}\right):$$

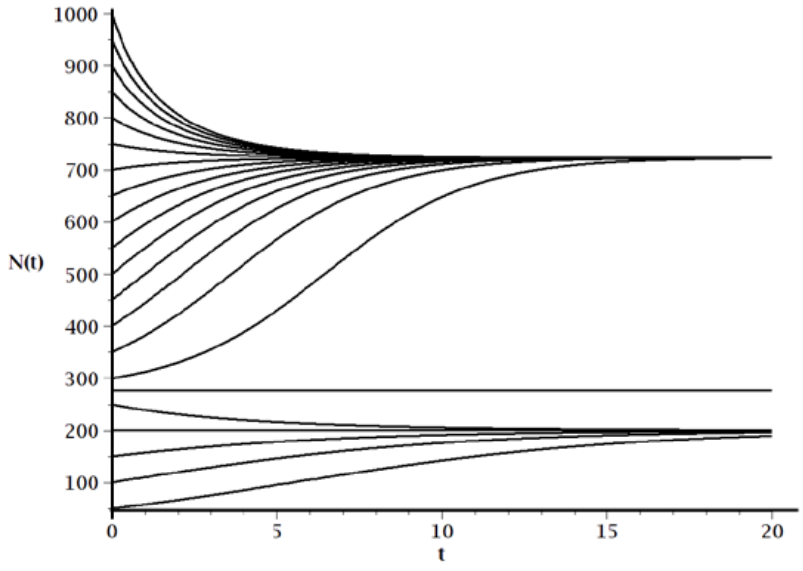
$$res[21] := dsolve\left(\left[\frac{d}{dt}N(t) = r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit} \cdot N(t) & N(t) \le Nkrit \end{cases}, N(0) = N0\right], numeric\right):$$

$$graf[21] := plots[odeplot](res[21], t = 0..20, thickness = 3, labels = ["t", "N(t)"], font = [Times, roman, 12], labelfont = [Times, bold, 12], numpoints = 1500):$$

Obrázek 3.22: Výpočet řešení modelu růstu populace pod tlakem nespecializovaného predátora v případě 2. b)

Vykreslení grafů:

$$plots[display](seq(graf[i], i = 1..21))$$



Obrázek 3.23: Vykreslení řešení modelu růstu populace pod tlakem nespecializovaného predátora v případě 2. b)

c) $N_{\text{krit}} \leq \frac{S}{r}$:

Nastavení parametrů: $S := 200$: $r := 1$: $K := 1000$: $Nkrit := 150$:

Cyklus vytvoření grafů pro jednotlivé hodnoty *N0*:
for i **from** 1 **to** 20 **do**
$N0 := i \cdot 50$:

$$res[i] := dsolve\left(\left[\frac{d}{dt}N(t) = r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit} \cdot N(t) & N(t) \leq Nkrit \end{cases}, N(0) = N0\right], numeric\right) :$$

$$graf[i] := plots[odeplot](res[i], t = 0..15, thickness = 3, labels = ["t [měsice]", "N(t) [jedinci]"], font = [Times, roman, 12], labelfont = [Times, bold, 12], numpoints = 1500) :$$

end do:

Vytvoření grafu pro hodnotu *N0* = N_2:

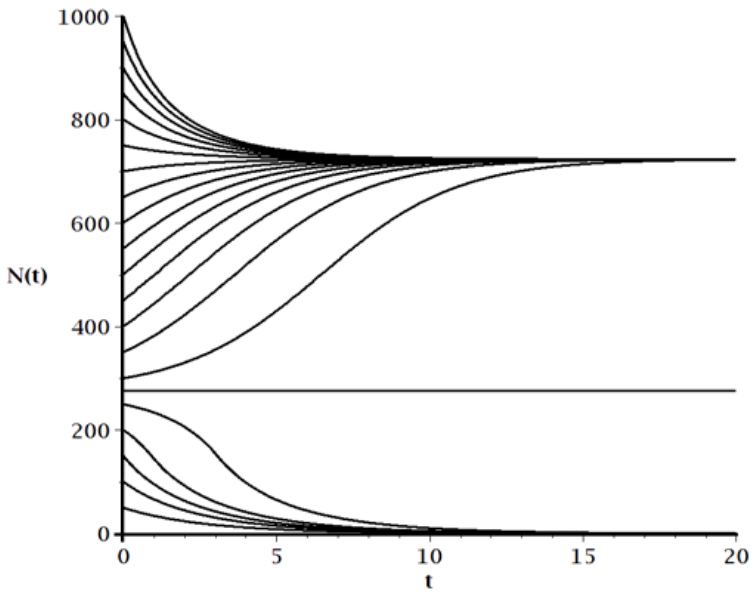
$$N0 := \frac{K}{2} \cdot \left(1 - \sqrt{1 - \frac{4 \cdot S}{r \cdot K}}\right) :$$

$$res[21] := dsolve\left(\left[\frac{d}{dt}N(t) = r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit} \cdot N(t) & N(t) \leq Nkrit \end{cases}, N(0) = N0\right], numeric\right) :$$

$$graf[21] := plots[odeplot](res[21], t = 0..15, thickness = 3, labels = ["t", "N(t)"], font = [Times, roman, 12], labelfont = [Times, bold, 12], numpoints = 1500) :$$

Obrázek 3.24: Řešení modelu růstu populace pod tlakem nespecializovaného predátora v případě 2. c)

Vykreslení grafů:
$plots[display](seq(graf[i], i = 1..21))$



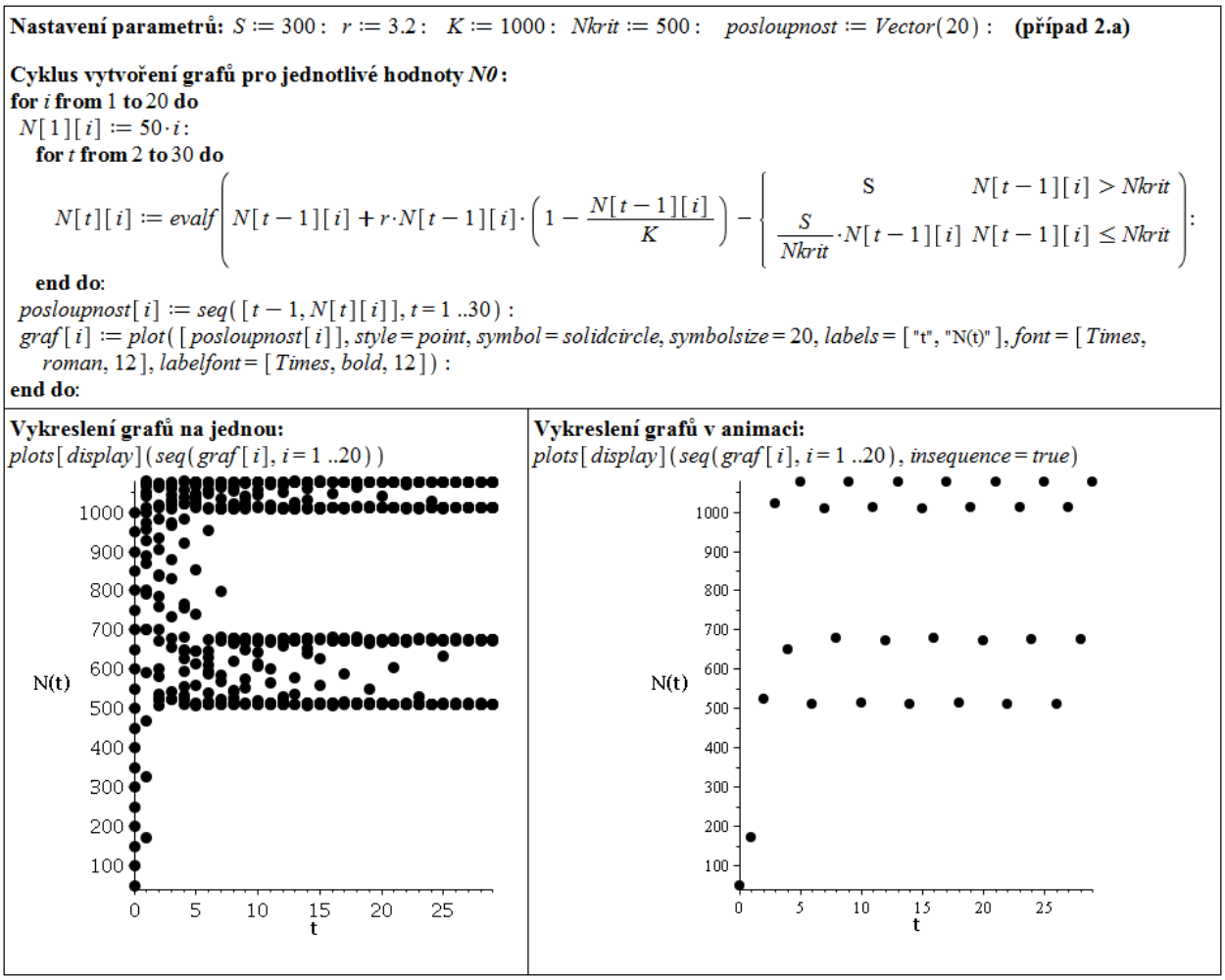
Obrázek 3.25: Řešení modelu růstu populace pod tlakem nespecializovaného predátora v případě 2. c)

Pokud $N0 > N_2$, velikost populace se ustálí na rovnovážné hodnotě N_3. Pokud $N0 < N_2$, ustálí se na hodnotě 0. Pro $N0 = N_2$ zůstane velikost populace konstantní. Predátor tedy zmenší ustálenou velikost populace nebo populaci vyhubí v závislosti na její počáteční velikosti – obrázky 3.24 a 3.25 pro nastavení parametrů:

$$S = 200,\ r = 1,\ K = 1000,\ N_{\text{krit}} = 400,\ N0 \in \{50, 100, ..., 1000\}$$

$$\Rightarrow N_2 \doteq 276, \quad N_3 \doteq 724$$

Předešlé možnosti chování modelu platí pro spojitý případ (3.16). V diskrétním případě (3.15) je situace složitější. Obrázek 3.26 ilustruje možnost 2. a) (tj. $S < \frac{1}{4} \cdot r \cdot K$, $N_{\text{krit}} > N_2$) v diskrétním případě. V levém grafu jsou vykreslena všechna řešení najednou, v pravém je vytvořena animace v závislosti na počáteční velikosti populace. Můžeme vidět, že v tomto případě se velikost populace v čase neustálí na jedné hodnotě, ale osciluje kolem ní.



Obrázek 3.26: Řešení modelu růstu populace pod tlakem nespecializovaného predátora v diskrétní podobě případu 2. a)

Při určitém nastavení parametrů se nám dokonce může stát, že řešení modelu bude dosahovat záporných čísel, jak ilustruje obrázek 3.27 pro možnost 1. a) (tj. $S \geq \frac{1}{4} \cdot r \cdot K$, $N_{\text{krit}} \leq \frac{S}{r}$). Příčinou je příliš velký časový krok pro dané hodnoty parametrů modelu.

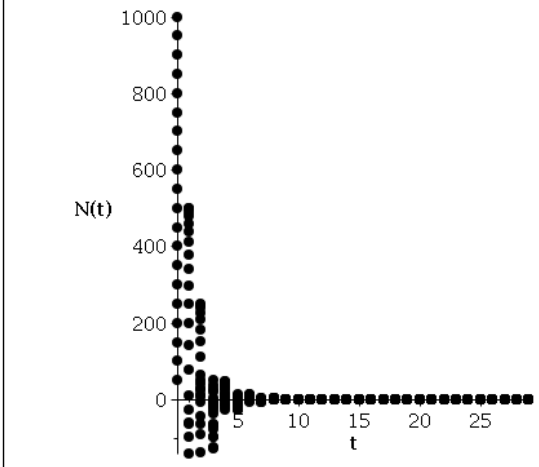
Nastavení parametrů: $S := 500:\ r := 1:\ K := 1000:\ Nkrit := 200:\ posloupnost := Vector(20):$ **(případ 1.a)**

Cyklus vytvoření grafů pro jednotlivé hodnoty *N0*:
for i **from** 1 **to** 20 **do**
$N[1][i] := 50 \cdot i:$
for t **from** 2 **to** 30 **do**

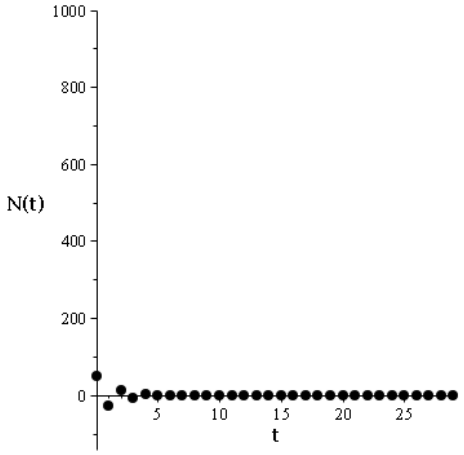
$$N[t][i] := evalf\left(N[t-1][i] + r \cdot N[t-1][i] \cdot \left(1 - \frac{N[t-1][i]}{K}\right) - \begin{cases} S & N[t-1][i] > Nkrit \\ \frac{S}{Nkrit} \cdot N[t-1][i] & N[t-1][i] \le Nkrit \end{cases}\right):$$

end do:
$posloupnost[i] := seq([t-1, N[t][i]], t = 1..30):$
$graf[i] := plot([posloupnost[i]], style = point, symbol = solidcircle, symbolsize = 20, labels = ["t", "N(t)"], font = [Times, roman, 12], labelfont = [Times, bold, 12]):$
end do:

Vykreslení grafů na jednou:
$plots[display](seq(graf[i], i = 1..20))$



Vykreslení grafů v animaci:
$plots[display](seq(graf[i], i = 1..20), insequence = true)$



Obrázek 3.27: Řešení modelu růstu populace pod tlakem nespecializovaného predátora v diskrétní podobě případu 1. a)

3.3 Interagující populace

Uvažujme obecně diskrétní rovnici (3.13) z předešlého modelu vyjadřující růst populace v omezeném prostředí, tj. rovnici

$$N(t+h) \;=\; N(t) \;+\; r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) \cdot h, \quad N(0) = N0 \tag{3.20}$$

a její diskrétní variantu

$$N(t+1) \;=\; N(t) \;+\; r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right), \quad N(0) = N0 \tag{3.21}$$

a spojitou variantu

$$N'(t) \;=\; r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right), \quad N(0) = N0. \tag{3.22}$$

Rovnici neomezeného (exponenciálního) růstu populace

$$N(t+h) \;=\; N(t) \;+\; r \cdot N(t) \cdot h, \tag{3.23}$$

případně její diskrétní variantu

$$N(t+1) \;=\; N(t) \;+\; r \cdot N(t) \tag{3.24}$$

nebo spojitou variantu

$$N'(t) \;=\; r \cdot N(t), \tag{3.25}$$

lze považovat za speciální případ rovnice (3.20), případně (3.21) nebo (3.22), pro $K = \infty$, neboť

$$\lim_{K \to \infty} \frac{N(t)}{K} = 0$$

pro jakoukoliv hodnotu $N(t)$.

Nyní představíme několik modelů dvou, případně tří, interagujících populací. Modely budeme prezentovat v diskrétní i spojité variantě, analýzu jejich řešení však budeme uvádět spolu s příklady v Maple pouze pro spojitý případ.

3.3.1 Modely dvou interagujících populací

Nyní budeme uvažovat dvě populace na jednom území. Označme $N_1(t)$, resp. $N_2(t)$, velikost první, resp. druhé, populace v čase t. Pokud by na sebe populace vzájemně nepůsobily, vývoj velikosti každé z nich by bylo možné modelovat pomocí rovnice (3.20); o kterou z populací jde, bychom odlišili dolním indexem u všech parametrů, tedy

$$N_1(t+h) \;=\; N_1(t) \;+\; r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t)}{K_1}\right) \cdot h,$$

$$N_2(t+h) \;=\; N_2(t) \;+\; r_2 \cdot N_2(t) \cdot \left(1 - \frac{N_2(t)}{K_2}\right) \cdot h.$$

Vliv velikosti jedné populace na růst druhé se může realizovat buď prostřednictvím prostředí, které obě populace obývají, nebo přímo.

Model, kdy j-tá populace ovlivňuje prostředí, v němž žije i-tá populace

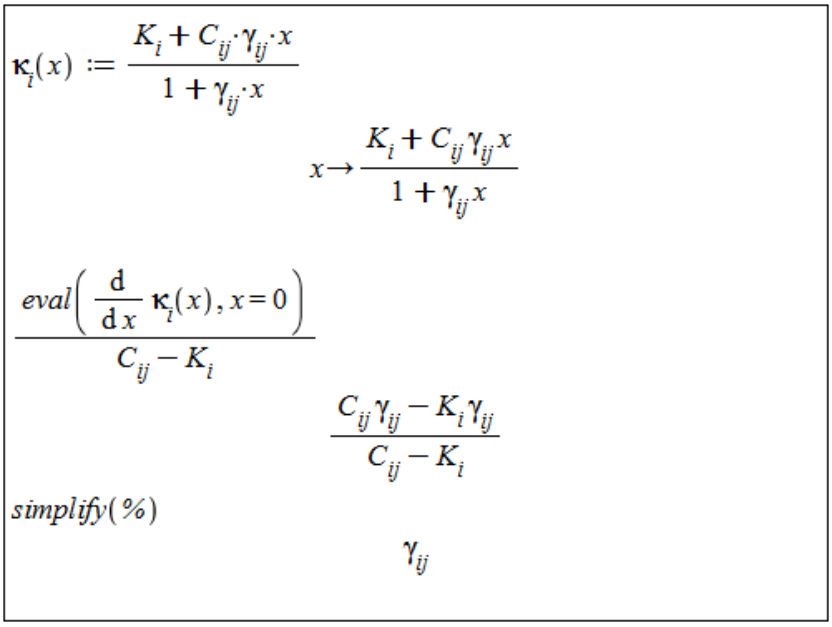
Nechť $i, j \in \{1, 2\}, i \neq j$. Kvalitu prostředí pro i-tou populaci v našem modelu vyjadřuje jediný parametr K_i – úživnost (nosná kapacita) prostředí. Pokud j-tá populace ovlivňuje prostředí, jeho úživnost již nebude konstanta K_i, ale bude záviset na velikosti j-té populace. Konstantu K_i nahradíme funkcí κ_i argumentu $N_j(t)$. Vývoj i-té populace tedy bude popsán rovnicí

$$N_i(t+h) \;=\; N_i(t) \;+\; r_i \cdot N_i(t) \cdot \left(1 - \frac{N_i(t)}{\kappa_i\big(N_j(t)\big)}\right) \cdot h. \tag{3.26}$$

Nyní budeme specifikovat funkci κ_i. Ta musí splňovat dva přirozené předpoklady:

- Pokud není j-tá populace přítomná, je úživnost prostředí nezměněna, tedy rovna původní hodnotě K_i. Přesněji
$$\kappa_i(0) = K_i.$$
- Pokud je j-tá populace veliká, změní úživnost prostředí na hodnotu C_{ij}, tedy
$$\lim_{N_j(t)\to\infty} \kappa_i\big(N_j(t)\big) = C_{ij}.$$

Jednoduchá funkce, která splňuje obě podmínky, je funkce lomená s lineární funkcí v čitateli i jmenovateli, tedy
$$\kappa_i\big(N_j(t)\big) = \frac{K_i + C_{ij}\cdot\gamma_{ij}\cdot N_j(t)}{1+\gamma_{ij}\cdot N_j(t)}. \tag{3.27}$$



Obrázek 3.28: Význam parametru γ_{ij}.

V předpisu funkce se objevuje nový parametr γ_{ij}. Abychom lépe pochopili jeho význam, určeme derivaci funkce κ_i v bodě 0 a výsledek vydělme rozdílem $C_{ij} - K_i$. Obrázek 3.28 ilustruje tento výpočet v systému Maple[21].

Parametr γ_{ij} vyjadřuje relativní změnu úživnosti prostředí pro i-tou populaci způsobenou malou (invazní) j-tou populací vzhledem k celkové možné změně úživnosti prostředí.

Je přirozené předpokládat, že konstanty K_i, C_{ij} jsou nezáporné (v prostředí nemůže být populace záporné velikosti) a že alespoň jedna z nich je kladná (prostředí populaci někdy uživí). Vztah konstant K_i a C_{ij} vyjadřuje ekologickou klasifikaci vztahu j-té populace k i-té:

- $K_i = C_{ij}$
  j-tá populace je k i-té *neutrální.*
- $K_i > C_{ij}$
  j-tá populace je *amensálem*[22] populace i-té.

[21] Pro názornost a možnost výpočtu v Maple bylo provedeno označení $x = N_j(t)$.

[22] *Amensalismus* je populační vztah, při němž jedna populace uvolňuje do prostředí odpadní produkt nebo speciální látku, která populaci jiného druhu ovlivňuje negativně (potlačuje růst a vývoj, způsobí i zánik) [18].

- $K_i < C_{ij}$
  j-tá populace je *komensálem*[23] populace i-té. V této situaci můžeme dále rozlišit:
  - $K_i = 0$ – i-tá populace by bez přítomnosti j-té nepřežila; j-tá populace je *obligátním komensálem* populace i-té.
  - $K_i > 0$ – i-tá populace přežívá i bez přítomnosti j-té; j-tá populace je *fakultativním komensálem* populace i-té.

Z rovností (3.26) a (3.27) dostáváme model vývoje velikostí dvou interagujících populací ve tvaru soustavy rovnic

$$\begin{aligned} N_1(t+h) &= N_1(t) + r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t) \cdot \big(1 + \gamma_{12} \cdot N_2(t)\big)}{K_1 + C_{12} \cdot \gamma_{12} \cdot N_2(t)}\right) \cdot h, \\ N_2(t+h) &= N_2(t) + r_2 \cdot N_2(t) \cdot \left(1 - \frac{N_2(t) \cdot \big(1 + \gamma_{21} \cdot N_1(t)\big)}{K_2 + C_{21} \cdot \gamma_{21} \cdot N_1(t)}\right) \cdot h. \end{aligned} \tag{3.28}$$

Tuto soustavu rovnic můžeme pro $h \to 0$ konkrétně zapsat jako soustavu rovnic diferenciálních

$$\begin{aligned} N_1'(t) &= r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t) \cdot \big(1 + \gamma_{12} \cdot N_2(t)\big)}{K_1 + C_{12} \cdot \gamma_{12} \cdot N_2(t)}\right), \\ N_2'(t) &= r_2 \cdot N_2(t) \cdot \left(1 - \frac{N_2(t) \cdot \big(1 + \gamma_{21} \cdot N_1(t)\big)}{K_2 + C_{21} \cdot \gamma_{21} \cdot N_1(t)}\right). \end{aligned} \tag{3.29}$$

nebo pro $h = 1$ jako soustavu rovnic diferenčních

$$\begin{aligned} N_1(t+1) &= N_1(t) + r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t) \cdot \big(1 + \gamma_{12} \cdot N_2(t)\big)}{K_1 + C_{12} \cdot \gamma_{12} \cdot N_2(t)}\right), \\ N_2(t+1) &= N_2(t) + r_2 \cdot N_2(t) \cdot \left(1 - \frac{N_2(t) \cdot \big(1 + \gamma_{21} \cdot N_1(t)\big)}{K_2 + C_{21} \cdot \gamma_{21} \cdot N_1(t)}\right). \end{aligned} \tag{3.30}$$

V případě komensalismu může dojít i k tomu, že populace komensála při rostoucí velikosti zvětšuje úživnost prostředí nade všechny meze, $\lim\limits_{N_j(t)\to\infty} \kappa_i\big(N_j(t)\big) = \infty$. Nejjednodušší funkce, která modeluje tento jev, je funkce lineární

$$\kappa_i\big(N_j(t)\big) = K_i + \gamma_{ij} \cdot N_j(t).$$

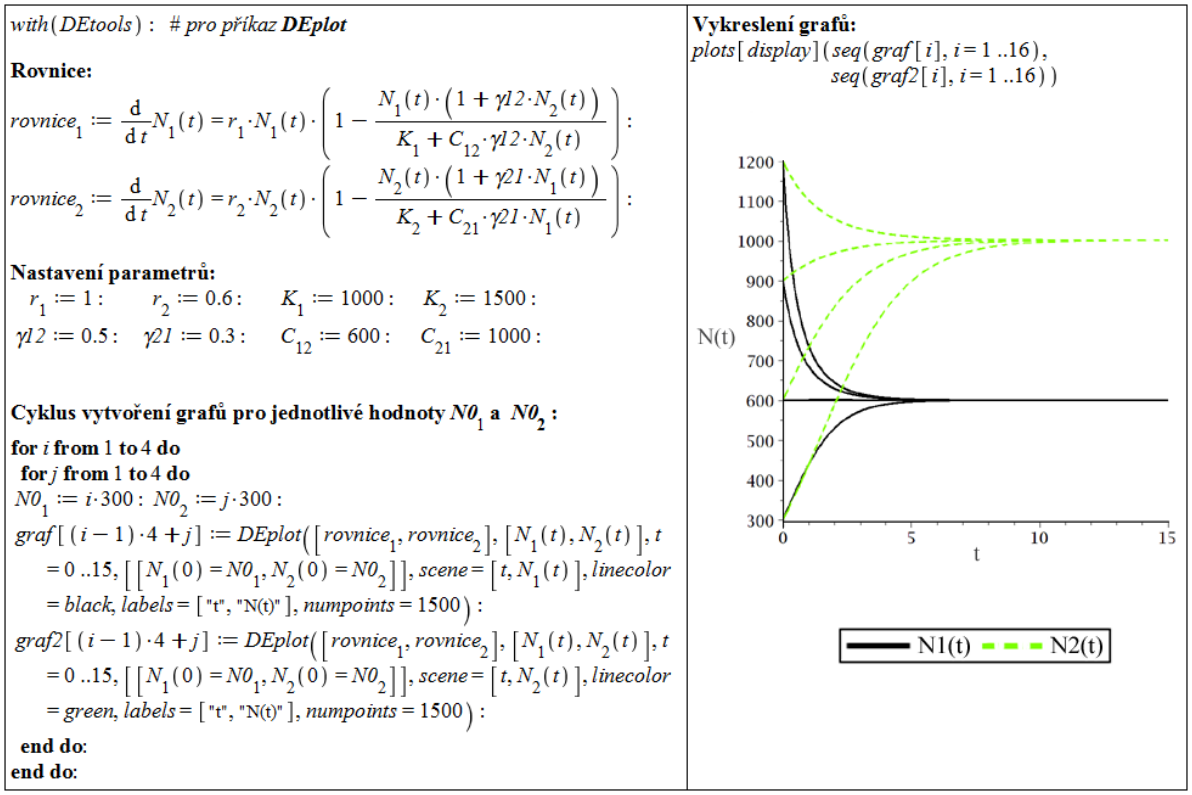
Zde kladný parametr γ_{ij} vyjadřuje absolutní nárůst úživnosti prostředí pro i-tou populaci způsobený j-tou populací o jednotkové velikosti.

Model konkurence (kompetice)

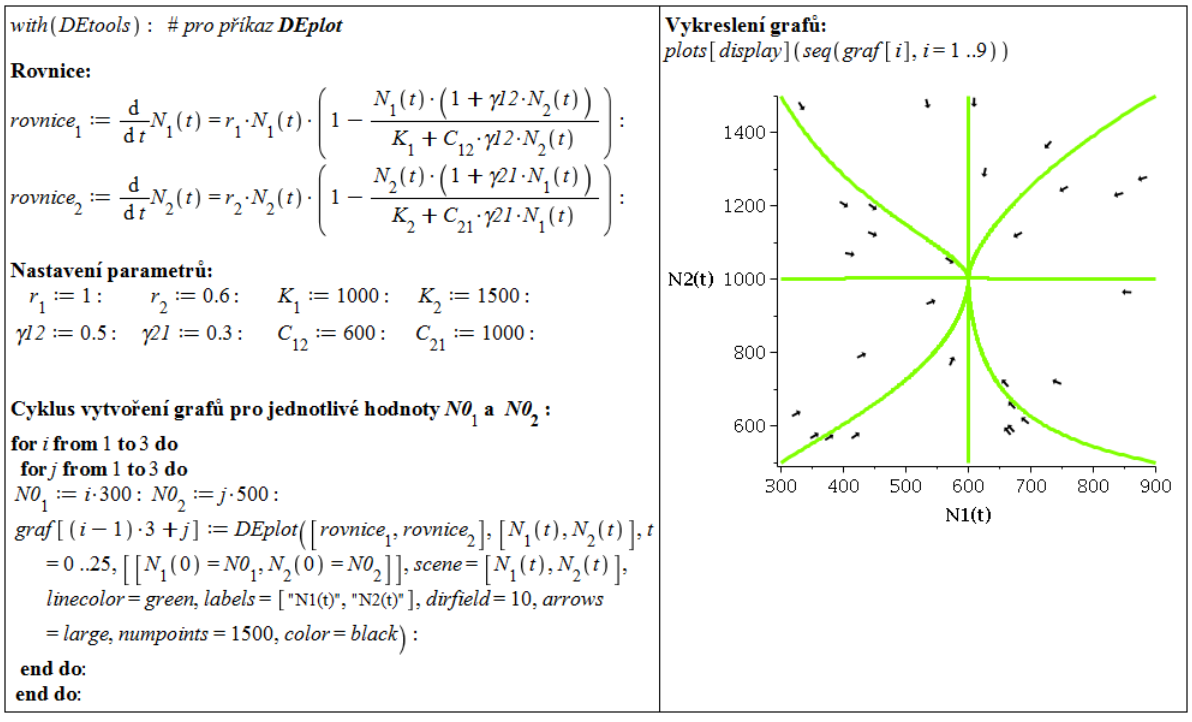
Model konkurence vyjadřuje interakci mezi populacemi způsobenou sdílenými požadavky na zdroj, který je omezeně dostupný [20]. Obě populace si tímto vzájemně snižují úživnost

[23] *Komensalismus* je populační vztah, při němž jedna populace využívá jinou bez jejího poškozování (jedna populace má ze vztahu prospěch, druhá není ovlivněna) [18].

prostředí. Velikosti obou populací se ustálí na nových rovnovážných hodnotách menších, než byly hodnoty pro izolované (vzájemně se neovlivňující) populace.



Obrázek 3.29: Řešení modelu konkurence dvou populací.



Obrázek 3.30: Řešení modelu konkurence dvou populací – směrové pole.

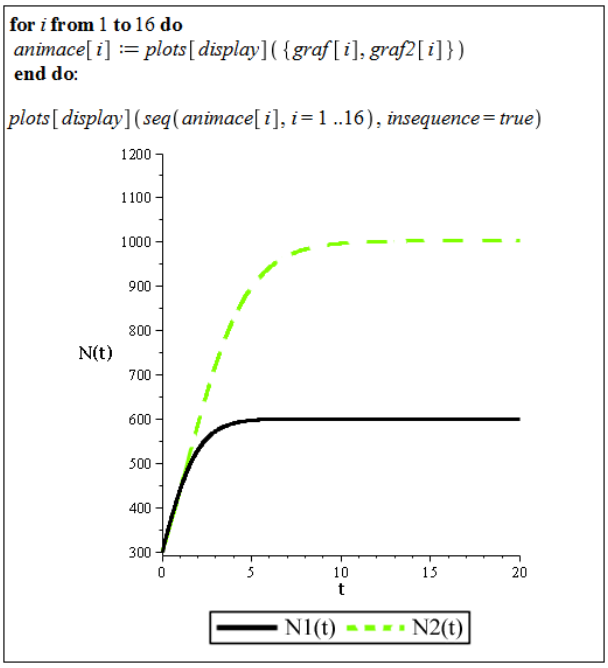
Obrázek 3.29 ukazuje příklad řešení modelu konkurence v Maple. Původní úživnosti prostředí jsou rovny 1000, resp. 1500, tj. v případě, že by se populace vzájemně neovlivňovaly (neovlivňovaly by prostředí, v nemž žijí), ustálily by se jejich velikosti na zmíněných hodnotách. Pro model konkurence platí $K_1 > C_{12}$ a $K_2 > C_{21}$. V příkladu na obrázku byly zvoleny hodnoty konstant $C_{12} = 600$ a $C_{21} = 1000$.

Řešení bylo vypočítáno pro různé počáteční velikosti populací $N0_1, N0_2 \in \{300, 600, 900, 1200\}$ tak, aby byly pokryty všechny vzájemné možnosti. Vidíme, že řešení nemá vliv na počáteční velikosti populací a vždy se velikosti obou populací ustálí na hodnotách C_{12} a C_{21}.

K zobrazení grafů jsme použili příkaz **DEplot**[24] k vykreslování řešení systému diferenciálních rovnic z balíku **DEtools**. Symbol γ je v systému Maple obsazený, parametr γ_{ij} proto zadáváme s indexy za řeckým písmenem (čímž vytvoříme neobsazený název pro proměnnou). Dvěma do sebe vnořenými **for**-cykly vytváříme grafy řešení pro různá nastavení počátečních hodnot.

Jedním z parametrů příkazu **DEplot** je atribut **scene**, v němž specifikujeme závislost, kterou chceme zobrazit. Ve všech grafech v tomto textu vykreslujeme závislost velikosti populace na čase. Nastavením parametru **scene** můžeme též zobrazit závislost velikosti jedné populace na velikosti druhé spolu se směrovým polem. Tuto možnost ilustruje obrázek 3.30.

Pro zobrazení jednotlivých řešení můžeme vytvořit animaci podobně jako u předchozího modelu v diskrétním případě. K programovému kódu z obrázku 3.29 stačí přidat kód z obrázku 3.31.



Obrázek 3.31: Řešení modelu konkurence dvou populací – animace.

Model mutualismu (symbiózy)

Symbióza se vyznačuje oboustranně kladným ovlivňováním dvou populací [18]. Obě populace si v tomto případě vzájemně zvyšují úživnost prostředí. V případě „omezeného ovlivňování“ modelovaného rovnicemi (3.29) nebo rovnicemi

[24]Příkaz **DEplot** nabízí volitelný parametr **numpoints** podobně jako příkaz **plot**. V tomto případě však bývá parametr **numpoints** standardně nastaven na hodnotu 49.

$$N_1'(t) \;=\; r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t) \cdot \left(1 + \gamma_{12} \cdot N_2(t)\right)}{K_1 + C_{12} \cdot \gamma_{12} \cdot N_2(t)}\right),$$

$$N_2'(t) \;=\; r_2 \cdot N_2(t) \cdot \left(1 - \frac{N_2(t)}{K_2 + \gamma_{21} \cdot N_1(t)}\right)$$

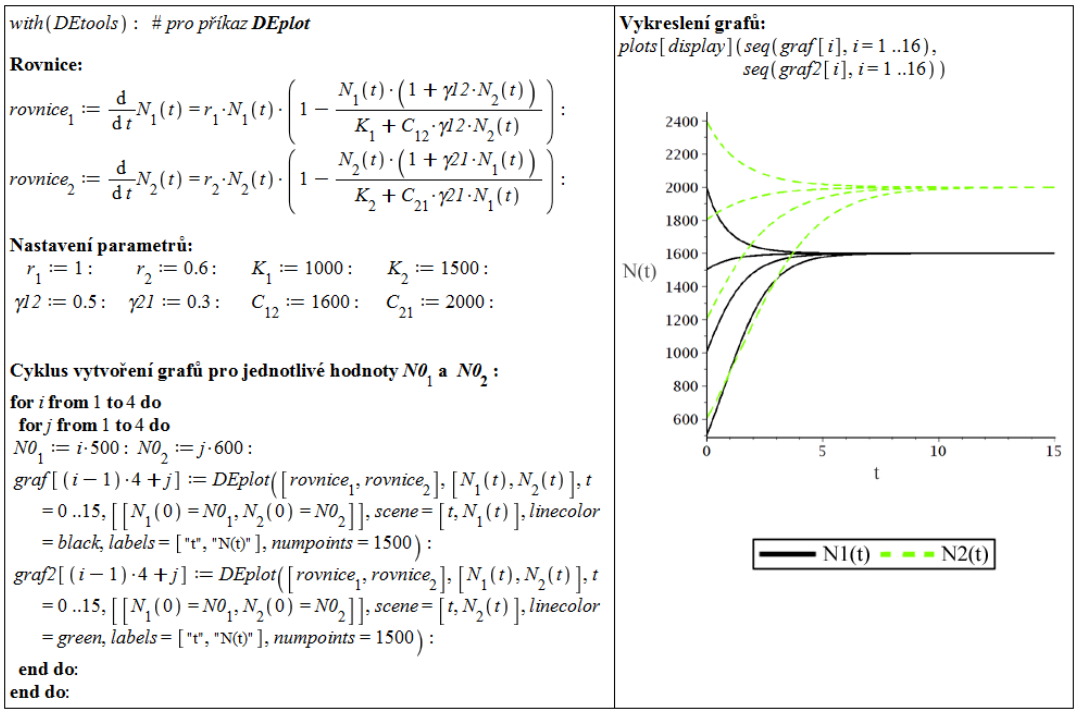
(alespoň jedna z konstant C_{12}, C_{21} je konečná) se velikosti obou populací ustálí na nových rovnovážných hodnotách. V případě „neomezeného ovlivňování“ modelovaného rovnicemi

$$N_1'(t) \;=\; r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t)}{K_1 + \gamma_{12} \cdot N_2(t)}\right),$$

$$N_2'(t) \;=\; r_2 \cdot N_2(t) \cdot \left(1 - \frac{N_2(t)}{K_2 + \gamma_{21} \cdot N_1(t)}\right)$$

záleží na hodnotách parametrů γ_{12}, γ_{21}. Pokud $\gamma_{12} \cdot \gamma_{21} < 1$, velikosti obou populací se ustálí na nových rovnovážných hodnotách. Pokud $\gamma_{12} \cdot \gamma_{21} \geq 1$, velikosti obou populací rostou nade všechny meze. V realitě by to znamenalo, že populace zničí svoje prostředí. Tomuto jevu se říká „orgie vzájemné dobročinnosti.“

Obrázky 3.32 – 3.34 ilustrují různé podoby chování řešení modelu symbiózy. Na prvním z nich je znázorněno řešení „omezeného ovlivňování“, kdy jsou obě konstanty C_{12}, C_{21} konečné a velikosti populací se ustálí na těchto hodnotách. Obrázek 3.33 ukazuje „neomezené ovlivňování“ v případě $\gamma_{12} \cdot \gamma_{21} < 1$, obrázek 3.34 v případě $\gamma_{12} \cdot \gamma_{21} \geq 1$.



Obrázek 3.32: Řešení modelu symbiózy dvou populací pro $C_{12} < \infty$, $C_{21} < \infty$.

with(DEtools) : # pro příkaz ***DEplot***

Rovnice:

$$rovnice_1 := \frac{d}{dt}N_1(t) = r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t)}{K_1 + \gamma 12 \cdot N_2(t)}\right) :$$

$$rovnice_2 := \frac{d}{dt}N_2(t) = r_2 \cdot N_2(t) \cdot \left(1 - \frac{N_2(t)}{K_2 + \gamma 21 \cdot N_1(t)}\right) :$$

Nastavení parametrů:

$r_1 := 1 :\quad r_2 := 0.6 :\quad K_1 := 1000 :\quad K_2 := 1500 :$

$\gamma 12 := 0.1 :\quad \gamma 21 := 0.7 :$

Cyklus vytvoření grafů pro jednotlivé hodnoty $N0_1$ a $N0_2$:

for *i* **from** 1 **to** 3 **do**
for *j* **from** 1 **to** 3 **do**

$N0_1 := i \cdot 500 :\quad N0_2 := j \cdot 900 :$

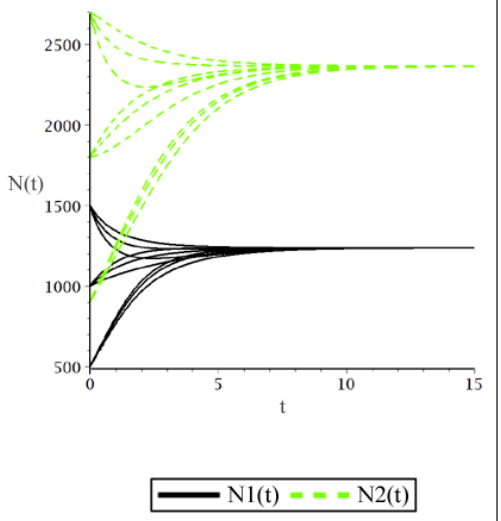
$graf[(i-1)\cdot 3 + j] := DEplot([rovnice_1, rovnice_2], [N_1(t), N_2(t)], t = 0..15, [[N_1(0) = N0_1, N_2(0) = N0_2]], scene = [t, N_1(t)], linecolor = black, labels = ["t", "N(t)"], numpoints = 1500) :$

$graf2[(i-1)\cdot 3 + j] := DEplot([rovnice_1, rovnice_2], [N_1(t), N_2(t)], t = 0..15, [[N_1(0) = N0_1, N_2(0) = N0_2]], scene = [t, N_2(t)], linecolor = green, labels = ["t", "N(t)"], numpoints = 1500) :$

end do:
end do:

Vykreslení grafů:

plots[display](seq(graf[i], i = 1..9), seq(graf2[i], i = 1..9))



Obrázek 3.33: Řešení modelu symbiózy dvou populací pro $C_{12} = C_{21} = \infty$, $\gamma_{12} \cdot \gamma_{21} < 1$.

with(DEtools) : # pro příkaz ***DEplot***

Rovnice:

$$rovnice_1 := \frac{d}{dt}N_1(t) = r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t)}{K_1 + \gamma 12 \cdot N_2(t)}\right) :$$

$$rovnice_2 := \frac{d}{dt}N_2(t) = r_2 \cdot N_2(t) \cdot \left(1 - \frac{N_2(t)}{K_2 + \gamma 21 \cdot N_1(t)}\right) :$$

Nastavení parametrů:

$r_1 := 1 :\quad r_2 := 0.6 :\quad K_1 := 1000 :\quad K_2 := 1500 :$

$\gamma 12 := 2.5 :\quad \gamma 21 := 1.3 :$

Cyklus vytvoření grafů pro jednotlivé hodnoty $N0_1$ a $N0_2$:

for *i* **from** 1 **to** 3 **do**

$N0_1 := 1000 \cdot i :\quad N0_2 := 1000 \cdot (4 - i) :$

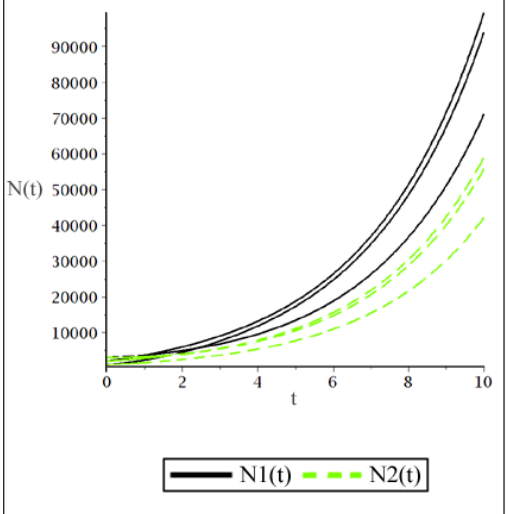
$graf[i] := DEplot([rovnice_1, rovnice_2], [N_1(t), N_2(t)], t = 0..10, [[N_1(0) = N0_1, N_2(0) = N0_2]], scene = [t, N_1(t)], linecolor = black, labels = ["t", "N(t)"], numpoints = 1500) :$

$graf2[i] := DEplot([rovnice_1, rovnice_2], [N_1(t), N_2(t)], t = 0..10, [[N_1(0) = N0_1, N_2(0) = N0_2]], scene = [t, N_2(t)], linecolor = green, labels = ["t", "N(t)"], numpoints = 1500) :$

end do:

Vykreslení grafů:

plots[display](seq(graf[i], i = 1..3), seq(graf2[i], i = 1..3))



Obrázek 3.34: Řešení modelu symbiózy dvou populací pro $C_{12} = C_{21} = \infty$, $\gamma_{12} \cdot \gamma_{21} \geq 1$.

Model predace

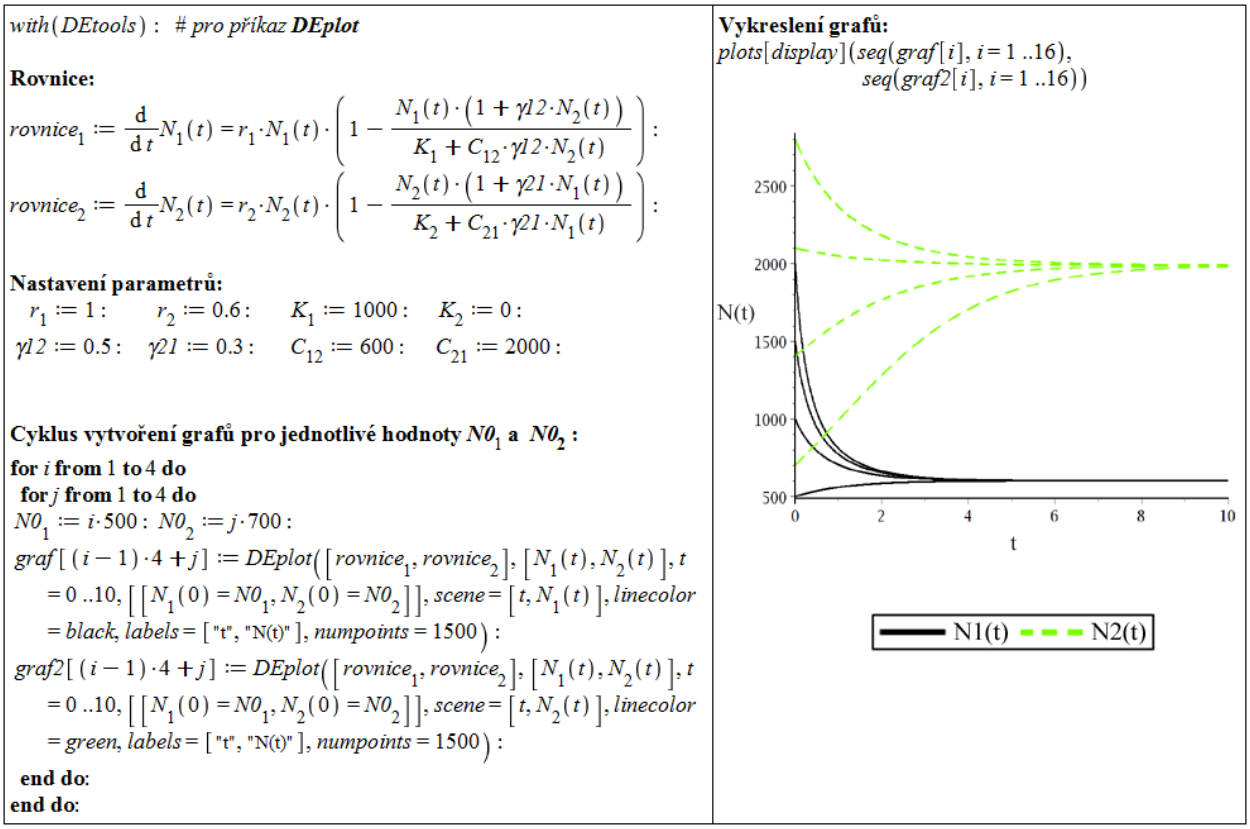
Obecně lze vztah predace definovat tak, že jedna populace (kořist) zvětšuje úživnost prostředí pro druhou populaci (predátora) a současně druhá populace zmenšuje růstový koeficient populace první. Zmenšení růstového koeficientu může být způsobeno jednak vlastní predací, tj. fyzickým hubením kořisti a tím zvětšením její úmrtnosti. Lze ho též modelovat jako zmenšení úživnosti prostředí a interpretovat tak, že kořist nemůže plně využívat zdroje prostředí, neboť se musí před predátory skrývat. Tento druhý pohled na predaci můžeme zformulovat jako model poulací ovlivňujících prostředí, v němž obě populace žijí. Přitom první populace (kořist) zvyšuje úživnost prostředí pro druhou populaci a druhá populace (predátor) naopak úživnost prostředí pro první populaci snižuje. Rozlišme dva případy:

- predátor je specializovaný – bez populace kořisti nemůže přežít ($K_2 = 0$),
- predátor je nespecializovaný – může přežít i bez populace kořisti, ale dostupnost populace kořisti ho ovlivňuje ($K_2 > 0$).

V obou případech se velikosti populací ustálí na nějaké rovnovážné hodnotě. U tohoto modelu predátor nemůže kořist vyhubit (na rozdíl od modelu 3.2, kde byla velikost populace predátora na populaci kořisti nezávislá) ani v případě, že zmenší kapacitu prostředí pro populaci kořisti na nulu, $C_{12} = 0$.

Alternativní modely predace jsou uvedeny na str. 53 a v podkapitolách 3.3.2, 3.3.3.

Obrázek 3.35 ukazuje řešení modelu predace v případě specializovaného predátora.



Obrázek 3.35: Řešení modelu predace pro $K_2 = 0$.

Modely, kdy j-tá populace ovlivňuje relativní přírustek i-té populace

Nechť stále $i, j \in \{1, 2\}, i \neq j$. Vývoj i-té populace, na niž působí populace j-tá, budeme modelovat rovnicí stejného typu, jako je rovnice (3.20). Vliv j-té populace na i-tou vyjádříme změnou relativního přírůstku i-té populace

$$r_i \cdot \left(1 - \frac{N_i(t)}{K_i}\right).$$

Budeme předpokládat, že j-tá populace, pokud s populací i-tou interaguje, k tomuto přírůstku přidává (nebo od něho ubírá) nějakou hodnotu. A dále, že tato hodnota je tím větší, čím větší je velikost j-té populace. Opět zvolíme tu nejjednodušší možnost: budeme předpokládat, že změna relativního přírůstku i-té populace je přímo úměrná velikosti populace j-té a konstantu úměrnosti označíme β_{ij}. Parametr β_{ij} budeme považovat za kladný, druh vlivu j-té populace na i-tou rozlišíme znaménkem. Je-li tedy j-tá populace komensálem populace i-té, bude vývoj i-té populace popsán rovnicí

$$\begin{aligned} N_i(t+h) \;&=\; N_i(t) \;+\; N_i(t) \cdot \left(r_i \cdot \left(1 - \frac{N_i(t)}{K_i}\right) + \beta_{ij} \cdot N_j(t)\right) \cdot h \;= \\ &=\; N_i(t) \;+\; N_i(t) \cdot \left(r_i - \frac{r_i}{K_i} \cdot N_i(t) + \beta_{ij} \cdot N_j(t)\right) \cdot h, \end{aligned}$$

je-li j-tá populace amensálem i-té, bude vývoj i-té populace popsán rovnicí

$$N_i(t+h) \;=\; N_i(t) \;+\; N_i(t) \cdot \left(r_i - \frac{r_i}{K_i} \cdot N_i(t) - \beta_{ij} \cdot N_j(t)\right) \cdot h.$$

Tento model lze jednotně zapsat ve tvaru

$$N_i(t+h) \;=\; N_i(t) \;+\; N_i(t) \cdot \left(r_i - \frac{r_i}{K_i} \cdot N_i(t) + s_{ij} \cdot \beta_{ij} \cdot N_j(t)\right) \cdot h,$$

kde

$$s_{ij} = \begin{cases} 1, & j\text{-tá populace je komensálem } i\text{-té} \\ -1, & j\text{-tá populace je amensálem } i\text{-té.} \end{cases}$$

Dynamika dvou populací tedy bude modelována soustavou dvou rovnic

$$\begin{aligned} N_1(t+h) \;&=\; N_1(t) \;+\; N_1(t) \cdot \left(r_1 - \frac{r_1}{K_1} \cdot N_1(t) + s_{12} \cdot \beta_{12} \cdot N_2(t)\right) \cdot h, \\ N_2(t+h) \;&=\; N_2(t) \;+\; N_2(t) \cdot \left(r_2 - \frac{r_2}{K_2} \cdot N_2(t) + s_{21} \cdot \beta_{21} \cdot N_1(t)\right) \cdot h. \end{aligned} \tag{3.31}$$

Pro $h \to 0$ dostaneme soustavu *Lotkových-Volterrových obyčejných diferenciálních rovnic*

$$\begin{aligned} N_1'(t) &= N_1(t) \cdot \left(r_1 - \frac{r_1}{K_1} \cdot N_1(t) + s_{12} \cdot \beta_{12} \cdot N_2(t) \right), \\ N_2'(t) &= N_2(t) \cdot \left(r_2 - \frac{r_2}{K_2} \cdot N_2(t) + s_{21} \cdot \beta_{21} \cdot N_1(t) \right). \end{aligned} \tag{3.32}$$

Model konkurence (kompetice)

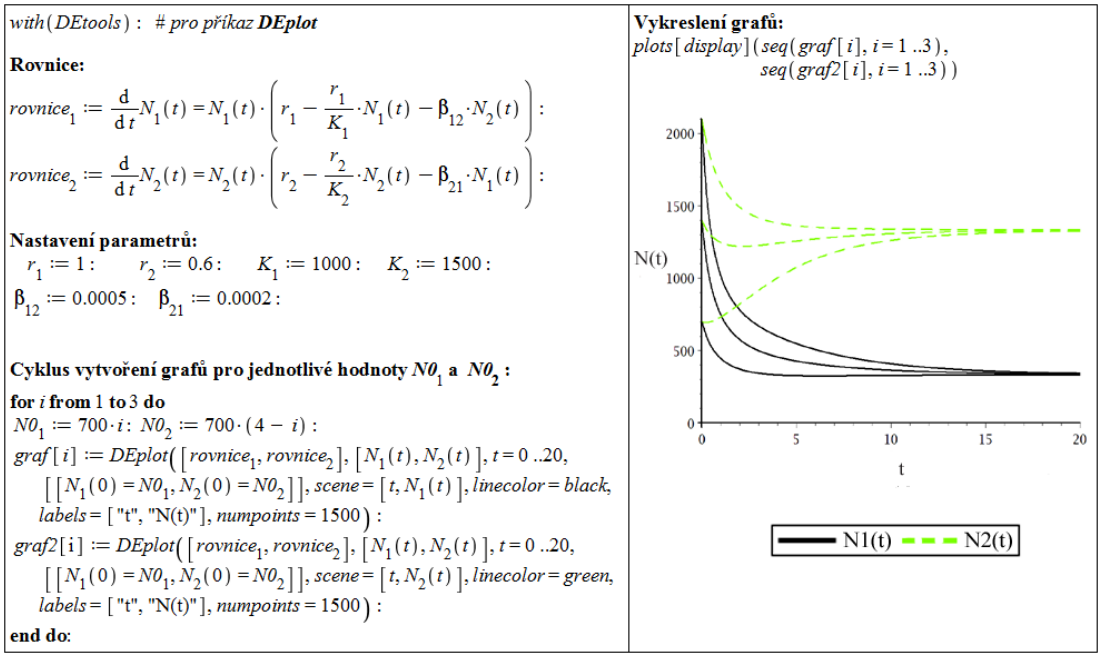
Model konkurence v tomto případě vyjadřuje situaci, kdy si obě populace vzájemně snižují své relativní přírůstky:

$$N_1'(t) = N_1(t) \cdot \left(r_1 - \frac{r_1}{K_1} \cdot N_1(t) - \beta_{12} \cdot N_2(t) \right),$$

$$N_2'(t) = N_2(t) \cdot \left(r_2 - \frac{r_2}{K_2} \cdot N_2(t) - \beta_{21} \cdot N_1(t) \right).$$

Na základě vztahů mezi parametry můžeme rozlišit čtyři různé kvalitativní výsledky modelu:

- $K_1 < \frac{r_2}{\beta_{21}}$, $K_2 < \frac{r_1}{\beta_{12}}$: velikosti populací se ustálí na hodnotách menších než původní úživnosti – *koexistence populací* (analogický výsledek k předchozímu modelu konkurence), obrázek 3.36.



Obrázek 3.36: Řešení modelu konkurence pro $K_1 < \frac{r_2}{\beta_{21}}$, $K_2 < \frac{r_1}{\beta_{12}}$.

- $K_1 < \frac{r_2}{\beta_{21}}$, $K_2 > \frac{r_1}{\beta_{12}}$: velikost druhé populace se ustálí na původní úživnosti prostředí, tj. na hodnotě K_2, velikost první populace se ustálí na hodnotě 0, tj. první populace vymře (*konkurenční vyloučení první populace*), obrázek 3.37.

with(DEtools) : # pro příkaz ***DEplot***

Rovnice:

$$rovnice_1 := \frac{d}{dt}N_1(t) = N_1(t)\cdot\left(r_1 - \frac{r_1}{K_1}\cdot N_1(t) - \beta_{12}\cdot N_2(t)\right):$$

$$rovnice_2 := \frac{d}{dt}N_2(t) = N_2(t)\cdot\left(r_2 - \frac{r_2}{K_2}\cdot N_2(t) - \beta_{21}\cdot N_1(t)\right):$$

Nastavení parametrů:

$r_1 := 1:\quad r_2 := 0.6:\quad K_1 := 1000:\quad K_2 := 1500:$

$\beta_{12} := 0.001:\quad \beta_{21} := 0.0002:$

Cyklus vytvoření grafů pro jednotlivé hodnoty $N0_1$ a $N0_2$:

for *i* **from** 1 **to** 3 **do**

$N0_1 := 700\cdot i:\ N0_2 := 700\cdot(4-i):$

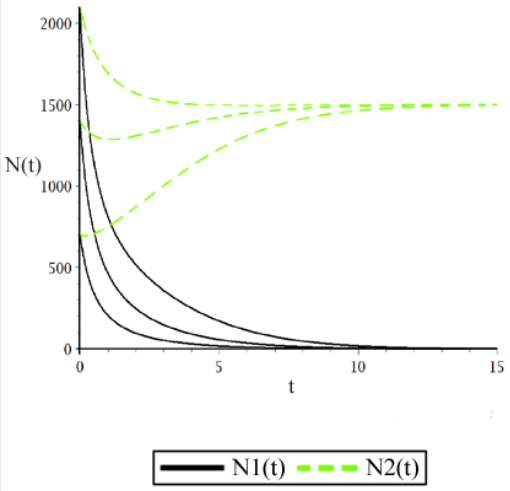
graf[i] := DEplot([rovnice1, rovnice2], [N1(t), N2(t)], t = 0..15, [[N1(0) = N01, N2(0) = N02]], scene = [t, N1(t)], linecolor = black, labels = ["t", "N(t)"], numpoints = 1500):

graf2[i] := DEplot([rovnice1, rovnice2], [N1(t), N2(t)], t = 0..15, [[N1(0) = N01, N2(0) = N02]], scene = [t, N2(t)], linecolor = green, labels = ["t", "N(t)"], numpoints = 1500):

end do:

Vykreslení grafů:

plots[display](seq(graf[i], i = 1..3), seq(graf2[i], i = 1..3))



Obrázek 3.37: Řešení modelu konkurence pro $K_1 < \frac{r_2}{\beta_{21}}$, $K_2 > \frac{r_1}{\beta_{12}}$.

with(DEtools) : # pro příkaz ***DEplot***

Rovnice:

$$rovnice_1 := \frac{d}{dt}N_1(t) = N_1(t)\cdot\left(r_1 - \frac{r_1}{K_1}\cdot N_1(t) - \beta_{12}\cdot N_2(t)\right):$$

$$rovnice_2 := \frac{d}{dt}N_2(t) = N_2(t)\cdot\left(r_2 - \frac{r_2}{K_2}\cdot N_2(t) - \beta_{21}\cdot N_1(t)\right):$$

Nastavení parametrů:

$r_1 := 1:\quad r_2 := 0.6:\quad K_1 := 1000:\quad K_2 := 1500:$

$\beta_{12} := 0.0006:\quad \beta_{21} := 0.003:$

Cyklus vytvoření grafů pro jednotlivé hodnoty $N0_1$ a $N0_2$:

for *i* **from** 1 **to** 3 **do**

$N0_1 := 700\cdot i:\ N0_2 := 700\cdot(4-i):$

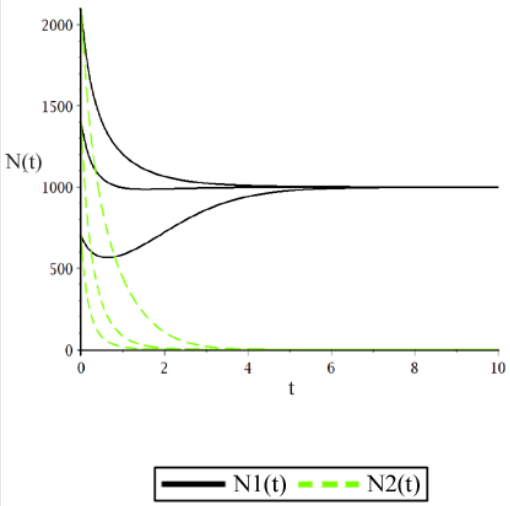
graf[i] := DEplot([rovnice1, rovnice2], [N1(t), N2(t)], t = 0..10, [[N1(0) = N01, N2(0) = N02]], scene = [t, N1(t)], linecolor = black, labels = ["t", "N(t)"], numpoints = 1500):

graf2[i] := DEplot([rovnice1, rovnice2], [N1(t), N2(t)], t = 0..10, [[N1(0) = N01, N2(0) = N02]], scene = [t, N2(t)], linecolor = green, labels = ["t", "N(t)"], numpoints = 1500):

end do:

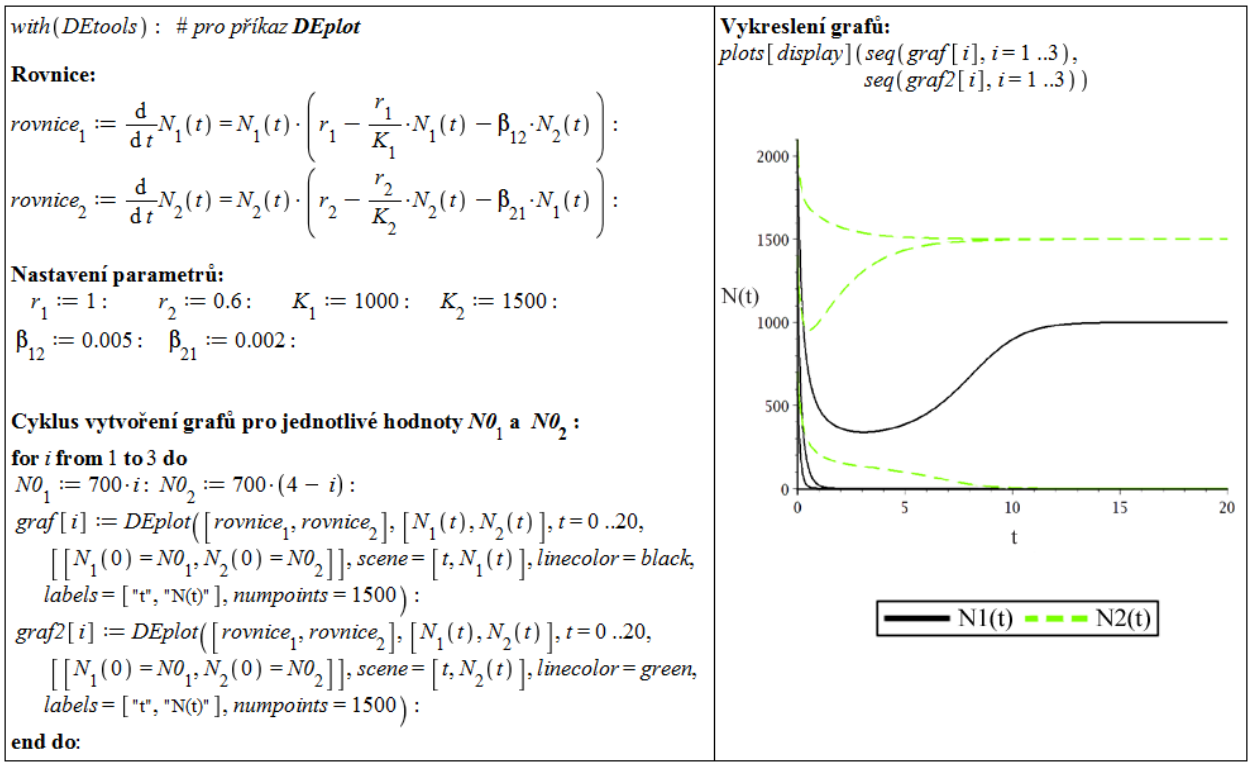
Vykreslení grafů:

plots[display](seq(graf[i], i = 1..3), seq(graf2[i], i = 1..3))



Obrázek 3.38: Řešení modelu konkurence pro $K_1 > \frac{r_2}{\beta_{21}}$, $K_2 < \frac{r_1}{\beta_{12}}$.

- $K_1 > \frac{r_2}{\beta_{21}}$, $K_2 < \frac{r_1}{\beta_{12}}$: velikost první populace se ustálí na původní úživnosti prostředí, tj. na hodnotě K_1, velikost druhé populace se ustálí na hodnotě 0, tj. druhá populace vymře (*konkurenční vyloučení druhé populace*), obrázek 3.38.
- $K_1 > \frac{r_2}{\beta_{21}}$, $K_2 > \frac{r_1}{\beta_{12}}$: jedna populace vymře, velikost zbývající se ustálí na původní úživnosti prostředí. Která populace vymře, závisí na počátečních hodnotách, jak dokumentuje obrázek 3.39. Pro zjištění, která z populací vymře a při jakých počátečních hodnotách jejich velikostí, můžeme opět využít animace zobrazující jednotlivá řešení, viz obrázek 3.40 .



Obrázek 3.39: Řešení modelu konkurence pro $K_1 > \frac{r_2}{\beta_{21}}$, $K_2 > \frac{r_1}{\beta_{12}}$.

Model predace

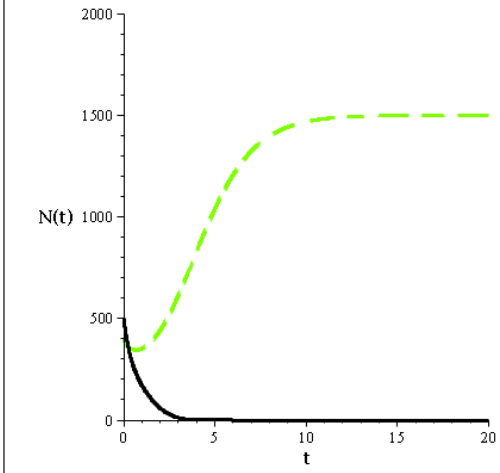
První populace (kořist) zvyšuje relativní přírůstek druhé populace (dravce), která první populaci hubí a tím její relativní přírůstek snižuje:

$$N_1'(t) = N_1(t) \cdot \left(r_1 - \frac{r_1}{K_1} \cdot N_1(t) - \beta_{12} \cdot N_2(t) \right),$$

$$N_2'(t) = N_2(t) \cdot \left(r_2 - \frac{r_2}{K_2} \cdot N_2(t) + \beta_{21} \cdot N_1(t) \right).$$

for i **from** 1 **to** 16 **do**
$animace[i] := plots[display](\{graf[i], graf2[i]\})$
end do:

$plots[display](seq(animace[i], i = 1..16), insequence = true)$



Obrázek 3.40: Animace řešení modelu konkurence pro $K_1 > \frac{r_2}{\beta_{21}}$, $K_2 > \frac{r_1}{\beta_{12}}$.

$with(DEtools)$: # *pro příkaz* ***DEplot***

Rovnice:

$$rovnice_1 := \frac{d}{dt}N_1(t) = N_1(t)\cdot\left(r_1 - \frac{r_1}{K_1}\cdot N_1(t) - \beta_{12}\cdot N_2(t)\right):$$

$$rovnice_2 := \frac{d}{dt}N_2(t) = N_2(t)\cdot\left(r_2 - \frac{r_2}{K_2}\cdot N_2(t) + \beta_{21}\cdot N_1(t)\right):$$

Nastavení parametrů:

$r_1 := 1$: $r_2 := 0.6$: $K_1 := 1000$: $K_2 := 1500$:

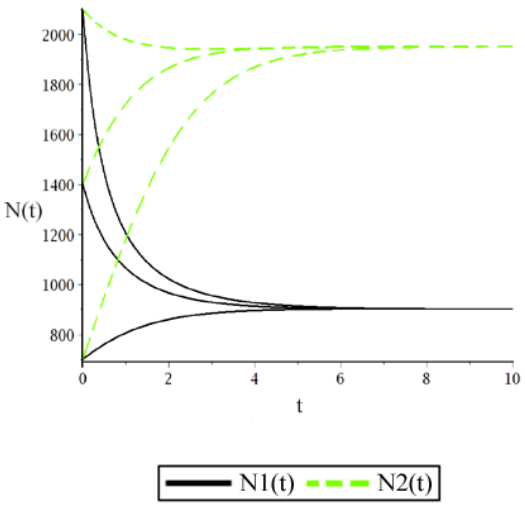
$\beta_{12} := 0.00005$: $\beta_{21} := 0.0002$:

Cyklus vytvoření grafů pro jednotlivé hodnoty $N0_1$ a $N0_2$:

for i **from** 1 **to** 3 **do**
$N0_1 := 700\cdot i$: $N0_2 := 700\cdot(4 - i)$:
$graf[i] := DEplot([rovnice_1, rovnice_2], [N_1(t), N_2(t)], t = 0..10, [[N_1(0) = N0_1, N_2(0) = N0_2]], scene = [t, N_1(t)], linecolor = black, labels = ["t", "N(t)"], numpoints = 1500)$:
$graf2[i] := DEplot([rovnice_1, rovnice_2], [N_1(t), N_2(t)], t = 0..10, [[N_1(0) = N0_1, N_2(0) = N0_2]], scene = [t, N_2(t)], linecolor = green, labels = ["t", "N(t)"], numpoints = 1500)$:
end do:

Vykreslení grafů:

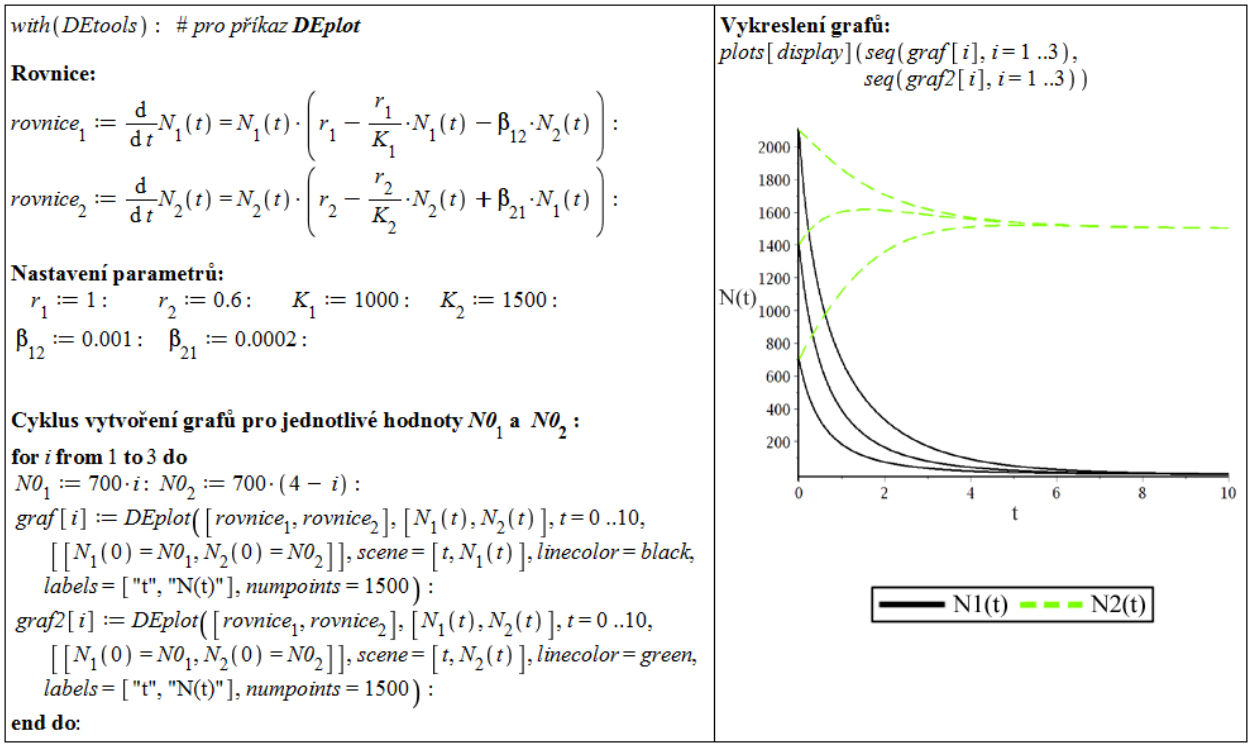
$plots[display](seq(graf[i], i = 1..3), seq(graf2[i], i = 1..3))$



Obrázek 3.41: Řešení modelu predace pro $K_2 < \frac{r_1}{\beta_{12}}$.

Poněvadž konstanta K_2 je ve jmenovateli, musí být nenulová, tj. $K_2 > 0$. Jedná se tedy o nespecializovaného predátora, který může přežít i bez uvažované populace kořisti, tj. má nějaké alternativní zdroje potravy.

Pokud $K_2 < \frac{r_1}{\beta_{12}}$, populace koexistují, jejich velikosti se ustálí na nějakých hodnotách – u populace kořisti menší než K_1, u populace dravce větší než K_2. Pokud $K_2 > \frac{r_1}{\beta_{12}}$, dravec kořist vyhubí. Popsané případy ilustrují obrázky 3.41 a 3.42.



Obrázek 3.42: Řešení modelu predace pro $K_2 > \frac{r_1}{\beta_{12}}$.

3.3.2 Model dravec-kořist Leslieho typu

Budeme předpokládat, že populace predátora zmenšuje relativní přírůstek populace kořisti a že populace kořisti zvětšuje úživnost prostředí pro populaci predátora. Velikost populace kořisti vlastně určuje velikost úživnosti prostředí pro populaci predátora. Pokud by tedy byla populace kořisti neomezená, byla by neomezená i úživnost. Za těchto předpokladů a při označení

$N_1(t)$ ... velikost populace kořisti v čase t,
$N_2(t)$ ... velikost populace predátora v čase t

dostáváme model vývoje velikostí populací ve tvaru

$$N_1(t+h) \;=\; N_1(t) \;+\; N_1(t)\cdot\left(r_1 - \frac{r_1}{K_1}\cdot N_1(t) - \alpha_{1,2}\cdot N_2(t)\right)\cdot h,$$

$$N_2(t+h) \;=\; N_2(t) \;+\; r_2\cdot N_2(t)\cdot\left(1 - \frac{N_2(t)}{K_2+\gamma_{21}\cdot N_1(t)}\right)\cdot h.$$

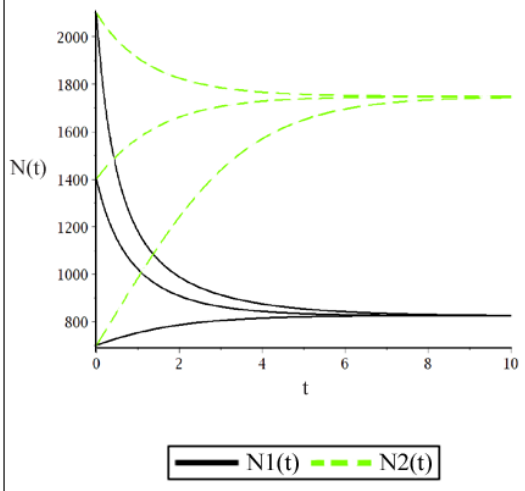
V případě $K_2 = 0$ se jedná o specializovaného predátora, v případě $K_2 > 0$ o nespecializovaného. Pro $h \to 0$ dostaneme soustavu diferenciálních rovnic[25]

$$N_1'(t) \;=\; N_1(t)\cdot\left(r_1 - \frac{r_1}{K_1}\cdot N_1(t) - \alpha_{1,2}\cdot N_2(t)\right),$$

$$N_2'(t) \;=\; r_2\cdot N_2(t)\cdot\left(1 - \frac{N_2(t)}{K_2+\gamma_{21}\cdot N_1(t)}\right).$$

```
with(DEtools) :  # pro příkaz DEplot

Rovnice:
rovnice1 := d/dt N1(t) = N1(t)·(r1 − r1/K1·N1(t) − α12·N2(t)) :
rovnice2 := d/dt N2(t) = r2·N2(t)·(1 − N2(t)/(K2 + γ21·N1(t))) :

Nastavení parametrů:
r1 := 1 :   r2 := 0.6 :   K1 := 1000 :   K2 := 1500 :
α12 := 0.0001 :   γ21 := 0.3 :

Cyklus vytvoření grafů pro jednotlivé hodnoty N0_1 a N0_2 :
for i from 1 to 3 do
N01 := 700·i :  N02 := 700·(4 − i) :
graf[i] := DEplot([rovnice1, rovnice2], [N1(t), N2(t)], t = 0..10,
    [[N1(0) = N01, N2(0) = N02]], scene = [t, N1(t)], linecolor = black,
    labels = ["t", "N(t)"], numpoints = 1500) :
graf2[i] := DEplot([rovnice1, rovnice2], [N1(t), N2(t)], t = 0..10,
    [[N1(0) = N01, N2(0) = N02]], scene = [t, N2(t)], linecolor = green,
    labels = ["t", "N(t)"], numpoints = 1500) :
end do:
```

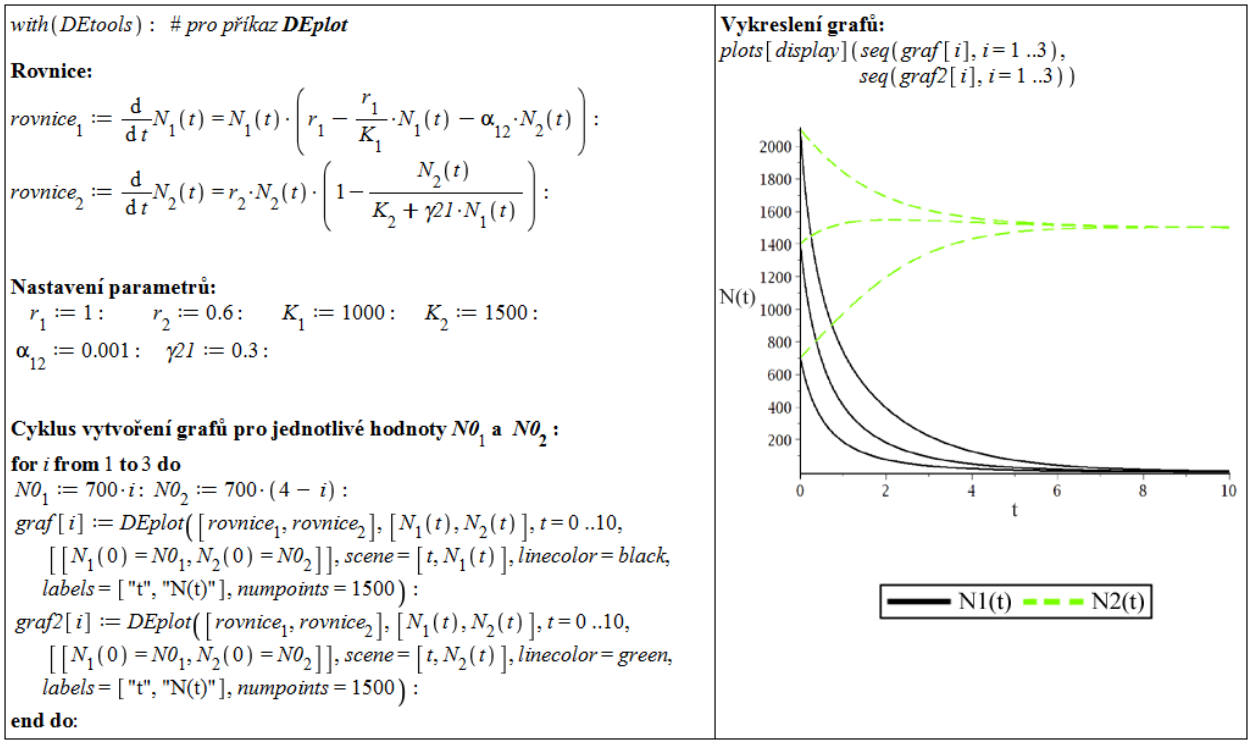
```
Vykreslení grafů:
plots[display](seq(graf[i], i = 1..3),
               seq(graf2[i], i = 1..3))
```



Obrázek 3.43: Řešení Leslieho modelu pro $K_2 < \frac{r_1}{\alpha_{12}}$.

Analogicky k předešlému modelu predace populace koexistují v případě $K_2 < \frac{r_1}{\alpha_{12}}$. V případě $K_2 > \frac{r_1}{\alpha_{12}}$ dravec kořist vyhubí (tato možnost může nastat jedině v případě nespecializovaného predátora, $K_2 > 0$).

[25] Model popsaný těmito diferenciálními rovnicemi nazýváme modelem dravec-kořist Leslieho typu [21].



Obrázek 3.44: Řešení Leslieho modelu pro $K_2 > \dfrac{r_1}{\alpha_{12}}$.

3.3.3 Model dravec-kořist Gauseho typu

Budeme předpokládat, že velikost populace kořisti, pokud by byla přítomná populace predátora o neměnné jednotkové velikosti, by se vyvíjela podle rovnice

$$N_1(t+h) \;=\; N_1(t) \;+\; r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t)}{K_1}\right) \cdot h \;-\; p\big(N_1(t)\big) \cdot h,$$

stejně jako populace pod konstantním tlakem nespecializovaného predátora (viz podkapitola 3.2). Výraz $p(N_1)$ vyjadřuje množství kořisti, které za jednotkový čas zničí populace predátora o určité velikosti. Tu můžeme považovat za jednotkovou, pokud má populace kořisti velikost $N_1(t)$. Funkce $p = p(N_1)$ se nazývá *trofická funkce* nebo *funkcionální odezva predátora na populaci kořisti o velikosti* N_1 [10].

Populace predátora o velikosti $N_2(t)$ tedy za jednotku času zničí $N_2 \cdot p(N_1)$ kořisti. Vývoj velikosti populace kořisti proto bude popsán rovnicí

$$N_1(t+h) \;=\; N_1(t) \;+\; r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t)}{K_1}\right) \cdot h \;-\; N_2(t) \cdot p\big(N_1(t)\big) \cdot h. \qquad (3.33)$$

O predátorovi budeme předpokládat, že je specializovaný (bez přítomnosti kořisti nemůže přežít). Vývoj velikosti jeho izolované populace by tedy bylo možno modelovat rovnicí (3.23) se záporným růstovým koeficientem, tj. rovnicí

$$N_2(t+h) \;=\; N_2(t) \;-\; \delta \cdot N_2(t) \cdot h.$$

Dále budeme předpokládat, že množství kořisti $N_2(t)\cdot p\big(N_1(t)\big)\cdot h$ zničené populací predátora za časový interval délky h se s nějakou efektivitou c přemění v populaci predátora. Z tohoto předpokladu dostaneme rovnici popisující vývoj velikosti populace predátora, který zničenou kořist transformuje do přírůstku své velikosti; tato rovnice je tvaru

$$N_2(t+h) \;=\; N_2(t) \;-\; \delta\cdot N_2(t)\cdot h \;+\; c\cdot N_2(t)\cdot p\big(N_1(t)\big)\cdot h. \tag{3.34}$$

Limitním přechodem $h\to 0$ dostaneme z rovnic (3.33) a (3.34) model vývoje velikostí populací kořisti a dravce jako soustavu dvou obyčejných diferenciálních rovnic[26]

$$\begin{aligned} N_1'(t) \;&=\; r_1\cdot N_1(t)\cdot\left(1-\frac{N_1(t)}{K_1}\right) \;-\; N_2(t)\cdot p\big(N_1(t)\big),\\ N_2'(t) \;&=\; N_2(t)\cdot\Big(-\delta + c\cdot p\big(N_1(t)\big)\Big). \end{aligned}$$

Ještě je třeba specifikovat trofickou funkci p. Můžeme použít stejnou funkci jako v modelu vývoje populace pod tlakem nespecializovaného predátora, tj.

$$p(N)=\begin{cases} \dfrac{S}{N_{\text{krit}}}\cdot N & \ldots\quad N\le N_{\text{krit}},\\[2ex] S & \ldots\quad N> N_{\text{krit}}. \end{cases} \tag{3.35}$$

Tato trofická funkce bývá v ekologické literatuře nazývána *Hollingova typu I*. Funkce podobného průběhu, tedy konkávní a taková, že $p(0)=0$ a $\lim\limits_{x\to\infty} p(x)=S$, avšak diferencovatelná (hladká) se nazývá *Hollingova typu II*. Nejjednodušší taková funkce je lomená

$$p(N_1)=S\cdot\frac{N_1}{N_1+\sigma};$$

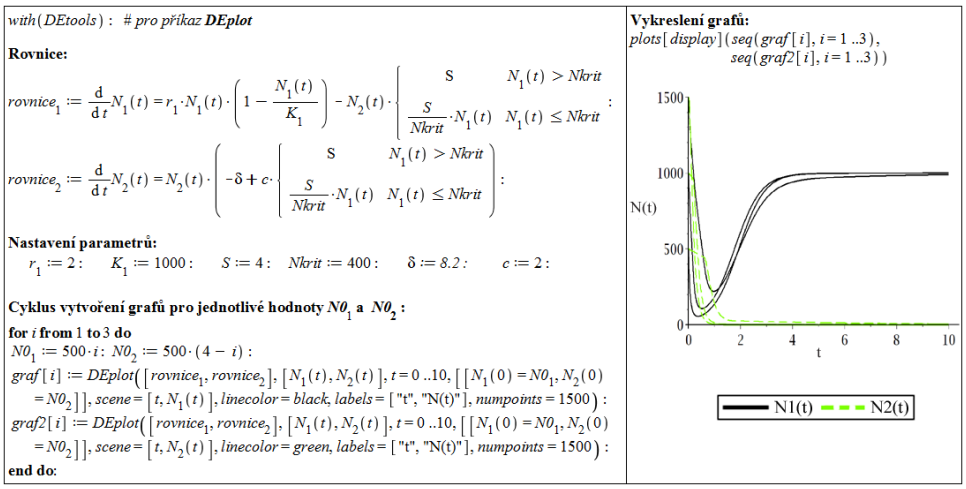
kladný parametr σ má podobný význam jako N_{krit}. Platí totiž

$$p'(0)=\frac{S}{N_{\text{krit}}}\text{ pro funkci typu I,}\quad p'(0)=\frac{S}{\sigma}\text{ pro funkci typu II.}$$

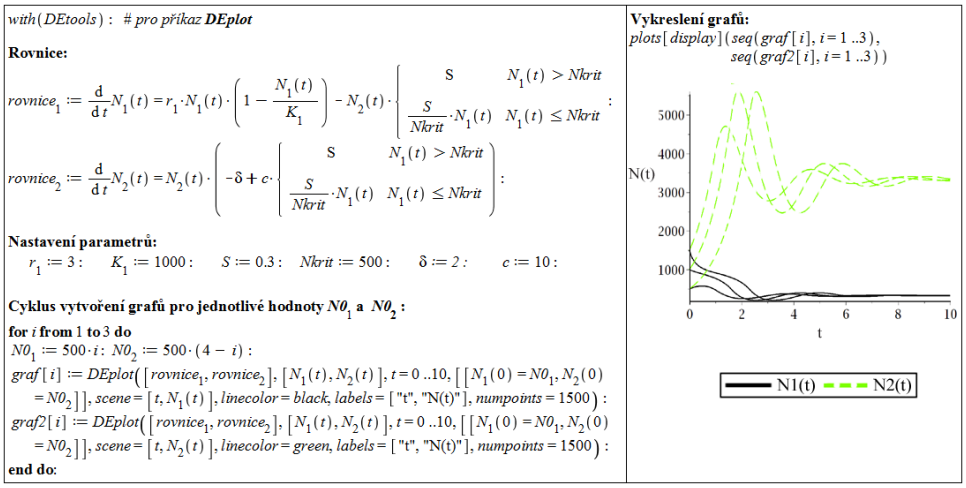
Nechť funkce p je nejprve trofická typu I. Obrázky 3.45 a 3.46 ukazují, že pro $\delta > c\cdot S$ vymře populace dravce a pro $\delta < c\cdot S$ obě populace koexistují při libovolných nenulových počátečních hodnotách.

V případě, že funkce p je trofická typu II, populace dravce opět vymře pro $\delta > c\cdot S$. Jestliže $\delta < c\cdot S$, obě populace koexistují tak, že se jejich velikosti ustálí, pokud navíc $\delta > c\cdot S\cdot\dfrac{K_1-\sigma}{K_1+\sigma}$, nebo kolísají, pokud $\delta < c\cdot S\cdot\dfrac{K_1-\sigma}{K_1+\sigma}$. Grafické znázornění zmíněných případů poskytují obrázky 3.47 – 3.49.

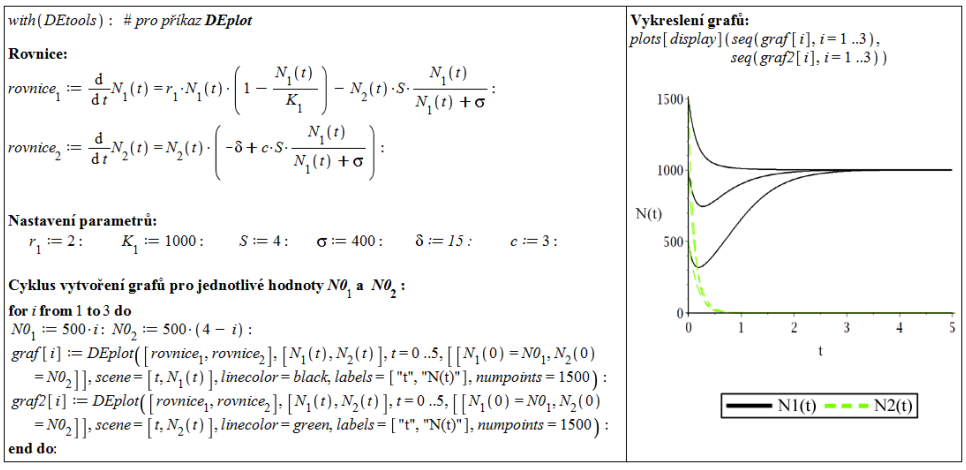
[26] Model popsaný těmito diferenciálními rovnicemi nazýváme modelem dravec-kořist Gauseho typu [8].



Obrázek 3.45: Řešení Gauseho modelu v případě, že p je trofická funkce typu I pro $\delta > c \cdot S$.



Obrázek 3.46: Řešení Gauseho modelu v případě, že p je trofická funkce typu I pro $\delta < c \cdot S$.



Obrázek 3.47: Řešení Gauseho modelu v případě, že p je trofická funkce typu II pro $\delta > c \cdot S$.

with(DEtools) : # pro příkaz **DEplot**

Rovnice:

$$rovnice_1 := \frac{d}{dt}N_1(t) = r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t)}{K_1}\right) - N_2(t) \cdot S \cdot \frac{N_1(t)}{N_1(t) + \sigma} :$$

$$rovnice_2 := \frac{d}{dt}N_2(t) = N_2(t) \cdot \left(-\delta + c \cdot S \cdot \frac{N_1(t)}{N_1(t) + \sigma}\right) :$$

Nastavení parametrů:

$r_1 := 2$: $K_1 := 1000$: $S := 2$: $\sigma := 400$: $\delta := 3.5$: $c := 3$:

Cyklus vytvoření grafů pro jednotlivé hodnoty $N0_1$ a $N0_2$:

for i **from** 1 **to** 3 **do**

$N0_1 := 500 \cdot i$: $N0_2 := 500 \cdot (4 - i)$:

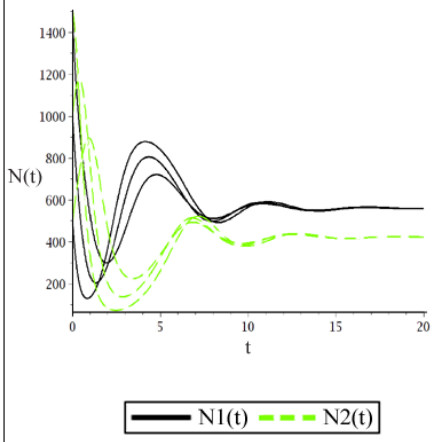
$graf[i] := DEplot([rovnice_1, rovnice_2], [N_1(t), N_2(t)], t = 0..20, [[N_1(0) = N0_1, N_2(0) = N0_2]], scene = [t, N_1(t)], linecolor = black, labels = ["t", "N(t)"], numpoints = 1500)$:

$graf2[i] := DEplot([rovnice_1, rovnice_2], [N_1(t), N_2(t)], t = 0..20, [[N_1(0) = N0_1, N_2(0) = N0_2]], scene = [t, N_2(t)], linecolor = green, labels = ["t", "N(t)"], numpoints = 1500)$:

end do:

Vykreslení grafů:

plots[display](seq(graf[i], i = 1..3), seq(graf2[i], i = 1..3))



Obrázek 3.48: Řešení Gauseho modelu v případě, že p je trofická funkce typu II pro $\delta < c \cdot S$ a $\delta > c \cdot S \cdot \frac{K_1 - \sigma}{K_1 + \sigma}$.

with(DEtools) : # pro příkaz **DEplot**

Rovnice:

$$rovnice_1 := \frac{d}{dt}N_1(t) = r_1 \cdot N_1(t) \cdot \left(1 - \frac{N_1(t)}{K_1}\right) - N_2(t) \cdot S \cdot \frac{N_1(t)}{N_1(t) + \sigma} :$$

$$rovnice_2 := \frac{d}{dt}N_2(t) = N_2(t) \cdot \left(-\delta + c \cdot S \cdot \frac{N_1(t)}{N_1(t) + \sigma}\right) :$$

Nastavení parametrů:

$r_1 := 2$: $K_1 := 1000$: $S := 2$: $\sigma := 400$: $\delta := 2.5$: $c := 3$:

Cyklus vytvoření grafů pro jednotlivé hodnoty $N0_1$ a $N0_2$:

for i **from** 1 **to** 3 **do**

$N0_1 := 200 \cdot i$: $N0_2 := 200 \cdot (4 - i)$:

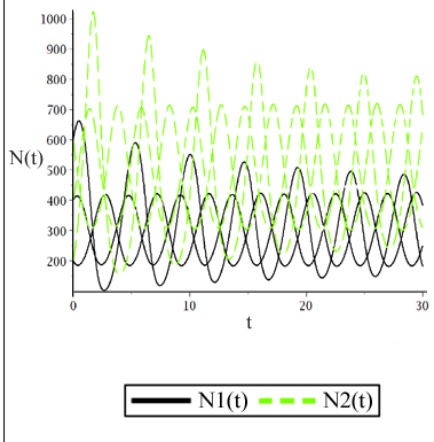
$graf[i] := DEplot([rovnice_1, rovnice_2], [N_1(t), N_2(t)], t = 0..30, [[N_1(0) = N0_1, N_2(0) = N0_2]], scene = [t, N_1(t)], linecolor = black, labels = ["t", "N(t)"], numpoints = 1500)$:

$graf2[i] := DEplot([rovnice_1, rovnice_2], [N_1(t), N_2(t)], t = 0..30, [[N_1(0) = N0_1, N_2(0) = N0_2]], scene = [t, N_2(t)], linecolor = green, labels = ["t", "N(t)"], numpoints = 1500)$:

end do:

Vykreslení grafů:

plots[display](seq(graf[i], i = 1..3), seq(graf2[i], i = 1..3))



Obrázek 3.49: Řešení Gauseho modelu v případě, že p je trofická funkce typu II pro $\delta < c \cdot S$ a $\delta < c \cdot S \cdot \frac{K_1 - \sigma}{K_1 + \sigma}$.

3.3.4 Společenstva n druhů – Lotkův-Volterrův systém

Opět budeme vycházet z modelu neomezeného růstu jedné populace ve tvaru

$$N(t + h) = N(t) + r \cdot N(t) \cdot h, \tag{3.36}$$

v jeho spojité

$$N'(t) = r \cdot N(t) \tag{3.37}$$

nebo diskrétní

$$N(t+1) = N(t) + r \cdot N(t) \tag{3.38}$$

variantě.

Uvažujme společenstvo tvořené n populacemi (biologickými druhy, případně vyššími či nižšími taxonomickými kategoriemi), které se mohou vzájemně ovlivňovat. Chceme modelovat vývoj velikostí jednotlivých populací v čase. Označíme

$N_i = N_i(t)$ ... velikost i-té populace v čase t,
r_i ... růstový koeficient i-té populace.

Vzájemné ovlivňování populací budeme modelovat tak, že růstový koeficient i-té populace r_i závisí na velikostech všech populací tvořících společenstvo (včetně i-té), tedy

$$r_i = r_i(N_1, N_2, \ldots, N_n), \quad i = 1, 2, \ldots, n.$$

Zvolíme tu nejjednodušší možnost, tedy závislost lineární,

$$r_i = a_i + \sum_{j=1}^{n} b_{ij} \cdot N_j.$$

Jednotlivé koeficienty lze interpretovat následovně:

a_i ... Vnitřní koeficient růstu i-té populace. Pokud $a_i > 0$, izolovaná i-tá populace by v daném prostředí rostla, pokud $a_i < 0$, izolovaná i-tá populace by v daném prostředí vymírala.

b_{ii} ... Síla vnitrodruhové konkurence nebo kooperace. Pokud $b_{ii} < 0$, jedná se o *vnitrodruhovou konkurenci*, pokud $b_{ii} > 0$, jedná se o *vnitrodruhovou kooperaci*.

b_{ij} ... Síla vlivu j-té populace na růst i-té.
$b_{ij} > 0$ ... j-tá populace je komensálem i-té,
$b_{ij} < 0$ ... j-tá populace je amensálem i-té,
$b_{ij} = 0$ ... j-tá populace je k i-té neutrální.

Tímto způsobem dostaneme model vývoje společenstva ve tvaru n rovnic

$$N_i(t+h) = N_i(t) + N_i(t) \cdot \left(a_i + \sum_{j=1}^{n} b_{ij} \cdot N_j(t) \right) \cdot h, \quad i = 1, 2, \ldots, n. \tag{3.39}$$

Obvykle je používán spojitý případ

$$N_i'(t) = N_i(t) \cdot \left(a_i + \sum_{j=1}^{n} b_{ij} \cdot N_j(t) \right), \quad i = 1, 2, \ldots, n, \tag{3.40}$$

známý jako *Lotkův-Volterrův systém*.

Například model vývoje dvou konkurujících si populací

$$N_1'(t) = N_1(t) \cdot \left(r_1 - \frac{r_1}{K_1} \cdot N_1(t) - \beta_{12} \cdot N_2(t) \right),$$

$$N_2'(t) = N_2(t) \cdot \left(r_2 - \frac{r_2}{K_2} \cdot N_2(t) - \beta_{21} \cdot N_1(t) \right).$$

je typu (3.40). V něm je

$$a_1 = r_1,\ b_{11} = -\frac{r_1}{K_1},\ b_{12} = -\beta_{12},\ a_2 = r_2,\ b_{21} = -\beta_{21},\ b_{22} = -\frac{r_2}{K_2}.$$

Povšimněme si, že za systém typu (3.40) lze považovat také model růstu jedné populace

$$N'(t) = r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K} \right); \tag{3.41}$$

zde je $n = 1$, $a_1 = r$, $a_{11} = -\frac{r}{K}$.

Někdy je výhodné, aby všechny parametry modelu byly kladné. V takovém případě položíme[27]

$$\alpha_i = |a_i|,\ s_i = \operatorname{sgn}(a_i),\ \beta_{ij} = |b_{ij}|,\ \sigma_{ij} = \operatorname{sgn}(b_{ij})$$

a systém (3.40) zapíšeme ve tvaru

$$N_i'(t) = N_i(t) \cdot \left(s_i \cdot \alpha_i + \sum_{j=1}^{n} \sigma_{ij} \cdot \beta_{ij} \cdot N_j(t) \right), \quad i = 1, 2, \ldots, n.$$

Nyní představíme tři různé modely soužití tří populací. Situace je ještě komplikovanější než dříve, modely obsahují více parametrů, a je tak více možností, jak mohou vypadat jejich řešení. U každého modelu si proto ukážeme jen nějaký zajímavý příklad.

Model konkurence tří populací

Model konkurence tří populací je popsán systémem rovnic

$$\begin{aligned} N_1'(t) &= N_1(t) \cdot \big(\alpha_1 - \beta_{11} \cdot N_1(t) - \beta_{12} \cdot N_2(t) - \beta_{13} \cdot N_3(t)\big), \\ N_2'(t) &= N_2(t) \cdot \big(\alpha_2 - \beta_{21} \cdot N_1(t) - \beta_{22} \cdot N_2(t) - \beta_{23} \cdot N_3(t)\big), \\ N_3'(t) &= N_3(t) \cdot \big(\alpha_3 - \beta_{31} \cdot N_1(t) - \beta_{32} \cdot N_2(t) - \beta_{33} \cdot N_3(t)\big). \end{aligned}$$

Nechť koeficienty splňují podmínky

$$\frac{\alpha_2}{\alpha_1}\frac{\beta_{12}}{\beta_{22}} > 1,\ \frac{\alpha_3}{\alpha_2}\frac{\beta_{23}}{\beta_{33}} > 1,\ \frac{\alpha_3}{\alpha_1}\frac{\beta_{31}}{\beta_{11}} > 1,\ \frac{\alpha_3}{\alpha_1}\frac{\beta_{13}}{\beta_{33}} < 1,\ \frac{\alpha_1}{\alpha_2}\frac{\beta_{21}}{\beta_{11}} < 1,\ \frac{\alpha_2}{\alpha_3}\frac{\beta_{32}}{\beta_{22}} < 1,$$

[27] **sgn** je znaménková funkce, tj. $\operatorname{sgn}(x) = 1$ pro $x > 0$, $\operatorname{sgn}(x) = -1$ pro $x < 0$, $\operatorname{sgn}(x) = 0$ pro $x = 0$.

$$\left(1-\frac{\alpha_3}{\alpha_1}\frac{\beta_{13}}{\beta_{33}}\right)\left(1-\frac{\alpha_1}{\alpha_2}\frac{\beta_{21}}{\beta_{11}}\right)\left(1-\frac{\alpha_2}{\alpha_3}\frac{\beta_{32}}{\beta_{22}}\right)>\left(\frac{\alpha_2}{\alpha_1}\frac{\beta_{12}}{\beta_{22}}-1\right)\left(\frac{\alpha_3}{\alpha_2}\frac{\beta_{23}}{\beta_{33}}-1\right)\left(\frac{\alpha_3}{\alpha_1}\frac{\beta_{31}}{\beta_{11}}-1\right).$$

Můžeme například volit

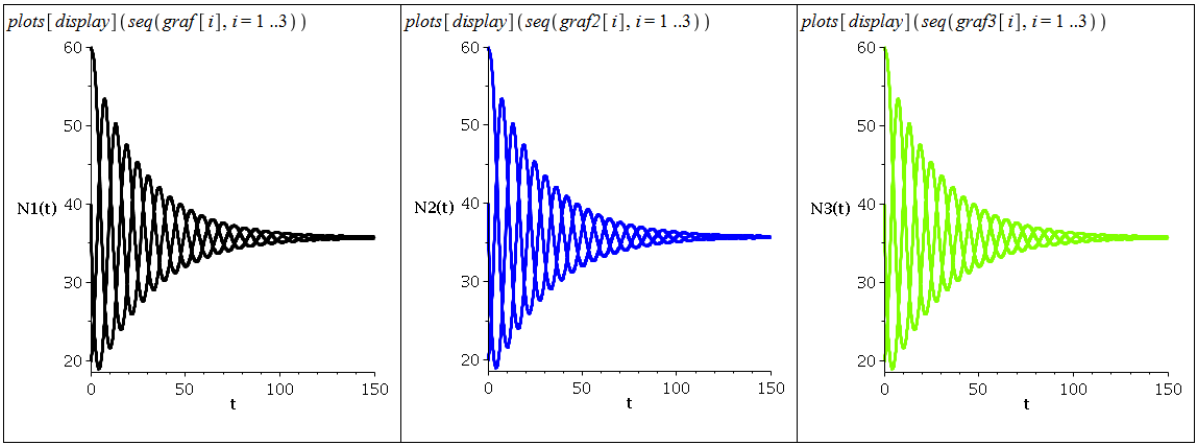
$$\alpha_1=\alpha_2=\alpha_3=1,\quad \beta_{11}=\beta_{22}=\beta_{33}=0.01,$$

$$\beta_{12}=\beta_{23}=\beta_{31}=0.015,\quad \beta_{21}=\beta_{32}=\beta_{13}=0.003.$$

V případě, že $N_1(0)>0$, $N_2(0)>0$, $N_3(0)>0$, se velikosti všech tří populací ustálí na nějaké nenulové hodnotě. Pokud je alespoň jedna z počátečních velikostí $N_1(0)$, $N_2(0)$, $N_3(0)$ nulová, pak přežije pouze jedna z populací tvořících společenstvo.

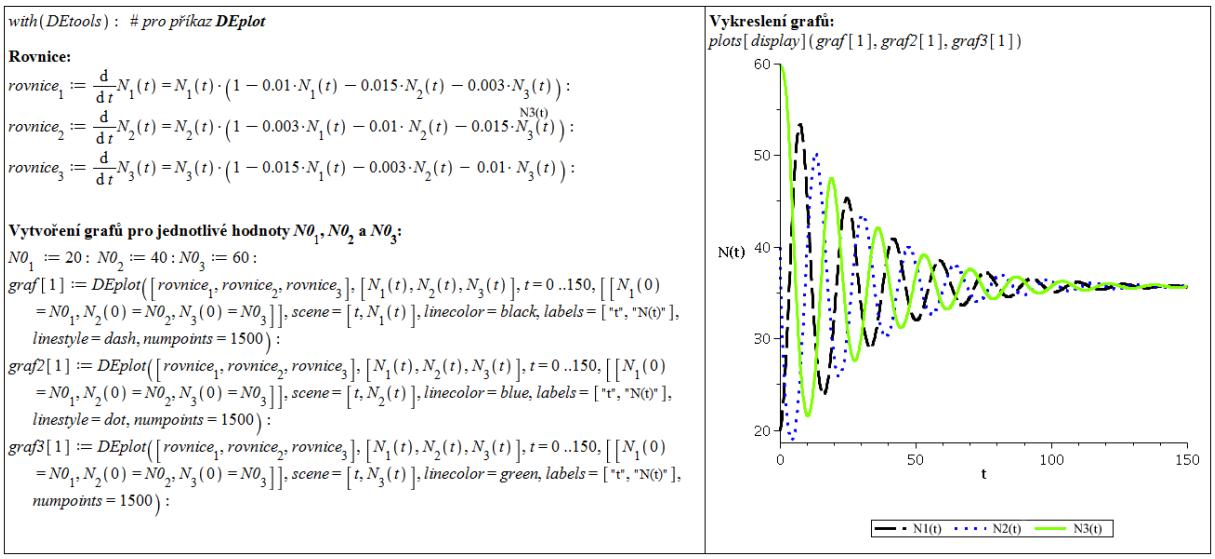
```
with(DEtools) :  # pro příkaz DEplot
Rovnice:
rovnice1 := (d/dt)N1(t) = N1(t)·(1 − 0.01·N1(t) − 0.015·N2(t) − 0.003·N3(t)) :
rovnice2 := (d/dt)N2(t) = N2(t)·(1 − 0.003·N1(t) − 0.01·N2(t) − 0.015·N3(t)) :
rovnice3 := (d/dt)N3(t) = N3(t)·(1 − 0.015·N1(t) − 0.003·N2(t) − 0.01·N3(t)) :

Cyklus vytvoření grafů pro jednotlivé hodnoty N01, N02 a N03:
poc_podm := [20, 40, 60, 20, 40] :

for i from 1 to 3 do
N01 := poc_podm[i] : N02 := poc_podm[i + 1] : N03 := poc_podm[i + 2] :
graf[i] := DEplot([rovnice1, rovnice2, rovnice3], [N1(t), N2(t), N3(t)], t = 0 ..150, [[N1(0) = N01, N2(0) = N02, N3(0) = N03]],
           scene = [t, N1(t)], linecolor = black, labels = ["t", "N1(t)"], numpoints = 1500) :
graf2[i] := DEplot([rovnice1, rovnice2, rovnice3], [N1(t), N2(t), N3(t)], t = 0 ..150, [[N1(0) = N01, N2(0) = N02, N3(0) = N03]],
           scene = [t, N2(t)], linecolor = blue, labels = ["t", "N2(t)"], numpoints = 1500) :
graf3[i] := DEplot([rovnice1, rovnice2, rovnice3], [N1(t), N2(t), N3(t)], t = 0 ..150, [[N1(0) = N01, N2(0) = N02, N3(0) = N03]],
           scene = [t, N3(t)], linecolor = green, labels = ["t", "N3(t)"], numpoints = 1500) :
end do:
```

Obrázek 3.50: Řešení modelu konkurence tří populací pro kladné počáteční hodnoty velikostí každé z nich – generující kód.

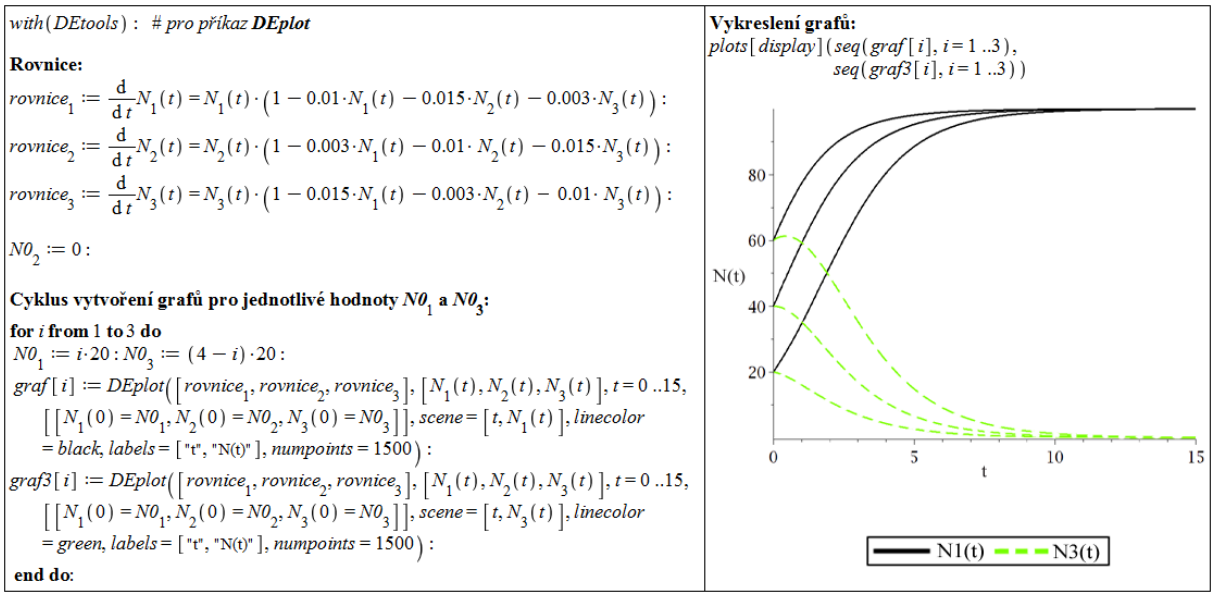


Obrázek 3.51: Zobrazení řešení modelu konkurence tří populací pro kladné počáteční hodnoty velikostí každé z nich – grafy.

Obrázky 3.50 a 3.51 ilustrují vytvoření grafů řešení modelu a jejich zobrazení pro různé nenulové počáteční velikosti populací. Grafy řešení jsou vykresleny pro každou populaci zvlášť. Můžeme pozorovat, že pro libovolnou počátení hodnotu se velikost populace vždy ustálí na stejné hodnotě. Obrázek 3.52 ukazuje pro jedno zvolené nastavení (každá populace má jinou počáteční velikost) velikosti všech tři populací v čase najednou. Případ, kdy jedna z populací není přítomna (tj. její počáteční velikost je nulová), je zobrazen na obrázku 3.53.



Obrázek 3.52: Řešení modelu konkurence tří populací pro kladné počáteční hodnoty velikostí každé z nich – jedno vybrané řešení.



Obrázek 3.53: Řešení modelu konkurence tří populací v případě nepřítomnosti jedné z nich.

Žádné dvě z konkurujících si populací nemohou koexistovat, všechny tři dohromady ano. Jedná se o příklad, kdy komplexnější společenstvo (tři konkurující si populace) je ekologicky stabilnější (populace dlouhodobě koexistují), než společenstvo méně komplexní (dvě konkurující si populace).

Predátor živící se dvěma konkurujícími si populacemi

Tento model popisuje systém rovnic

$$\begin{aligned} N_1'(t) &= N_1(t)\cdot\big(\alpha_1 - \beta_{11}\cdot N_1(t) - \beta_{12}\cdot N_2(t) - \beta_{13}\cdot N_3(t)\big),\\ N_2'(t) &= N_2(t)\cdot\big(\alpha_2 - \beta_{21}\cdot N_1(t) - \beta_{22}\cdot N_2(t) - \beta_{23}\cdot N_3(t)\big),\\ N_3'(t) &= N_3(t)\cdot\big(-\alpha_3 + \beta_{31}\cdot N_1(t) + \beta_{32}\cdot N_2(t)\big). \end{aligned}$$

V tomto případě $N_1 = N_1(t)$ a $N_2 = N_2(t)$ označují velikosti konkurujících si populací kořisti, $N_3 = N_3(t)$ označuje velikost populace predátora.

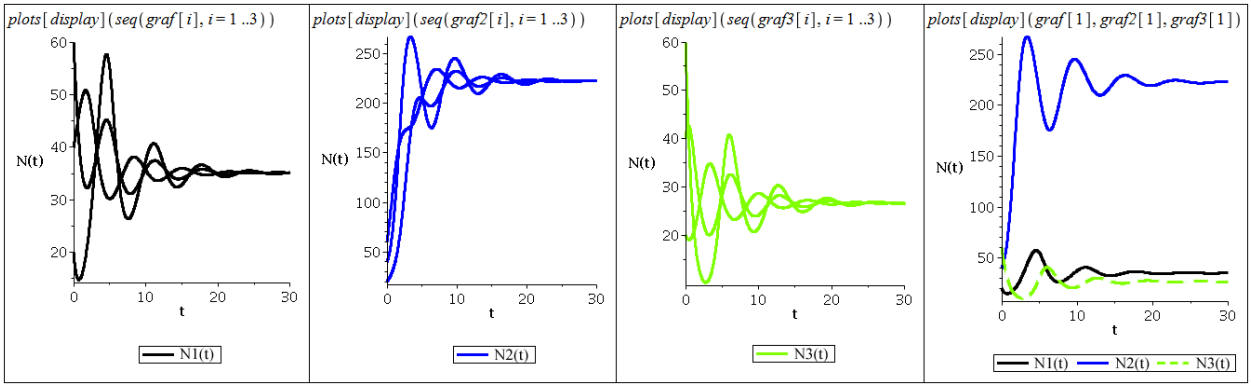
Zajímavá volba parametrů může být

$$\alpha_1 = 1,\ \alpha_2 = 2,\ \alpha_3 = 1.5,\quad \beta_{11} = 0.005,\ \beta_{12} = 0.0001,\ \beta_{13} = 0.03,$$

$$\beta_{21} = 0.01,\ \beta_{22} = 0.005,\ \beta_{23} = 0.02,\quad \beta_{31} = 0.03,\ \beta_{32} = 0.002.$$

```
with(DEtools) :  # pro příkaz DEplot

Rovnice:
rovnice1 := d/dt N1(t) = N1(t)·(1 − 0.005·N1(t) − 0.0001·N2(t) − 0.03·N3(t)) :
rovnice2 := d/dt N2(t) = N2(t)·(2 − 0.01·N1(t) − 0.005·N2(t) − 0.02·N3(t)) :
rovnice3 := d/dt N3(t) = N3(t)·(−1.5 + 0.03·N1(t) + 0.002·N2(t)) :

Cyklus vytvoření grafů pro jednotlivé hodnoty N0[1], N0[2] a N0[3]:
pocatecni_podminky := [20, 40, 60, 20, 40] :
for i from 1 to 3 do
 N0[1] := pocatecni_podminky[i] : N0[2] := pocatecni_podminky[i + 1] : N0[3] := pocatecni_podminky[i + 2] :
 graf[i] := DEplot([rovnice1, rovnice2, rovnice3], [N1(t), N2(t), N3(t)], t = 0..30, [[N1(0) = N0[1], N2(0) = N0[2], N3(0) = N0[3]]], scene = [t, N1(t)], linecolor = black, labels
    = ["t", "N(t)"], numpoints = 1500) :
 graf2[i] := DEplot([rovnice1, rovnice2, rovnice3], [N1(t), N2(t), N3(t)], t = 0..30, [[N1(0) = N0[1], N2(0) = N0[2], N3(0) = N0[3]]], scene = [t, N2(t)], linecolor = blue, labels
    = ["t", "N(t)"], numpoints = 1500) :
 graf3[i] := DEplot([rovnice1, rovnice2, rovnice3], [N1(t), N2(t), N3(t)], t = 0..30, [[N1(0) = N0[1], N2(0) = N0[2], N3(0) = N0[3]]], scene = [t, N3(t)], linecolor = green, labels
    = ["t", "N(t)"], numpoints = 1500) :
 end do:
```

Obrázek 3.54: Řešení modelu predátora živícího se dvěma konkurujícími si populacemi za přítomnosti predátora – generující kód.

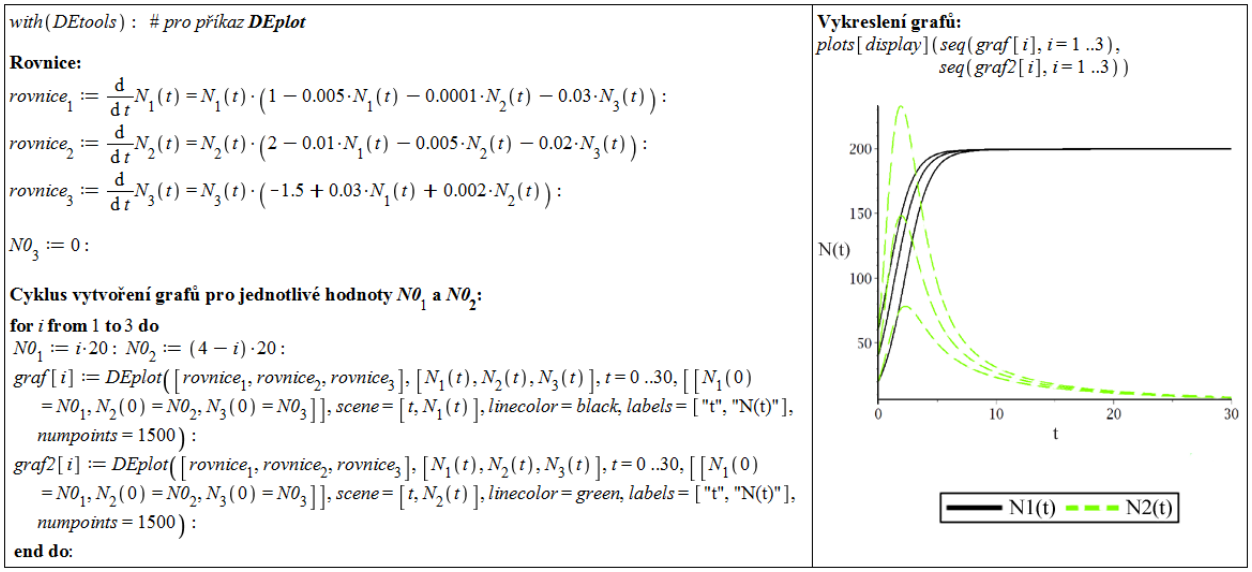


Obrázek 3.55: Řešení modelu predátora živícího se dvěma konkurujícími si populacemi za přítomnosti predátora – grafy řešení .

Pokud $N_3(0) > 0$ (populace predátora je ve společenstvu přítomná), je $\lim_{t\to\infty} N_1(t) > 0$ a také $\lim_{t\to\infty} N_2(t) > 0$, tj. obě konkurující si populace přežívají – obrázky 3.54 a 3.55). Zobrazena jsou opět jednotlivá řešení, poslední z grafů ukazuje všechna tři řešení najednou (každá populace má jinou počáteční velikost).

Pokud $N_3(0) = 0$ (ve společenstvu se nevyskytuje predátor), pak při jakékoliv volbě počátečních velikostí populací kořisti takových, že $N_1(0) > 0$ je $\lim_{t\to\infty} N_2(t) = 0$ (druhá z populací kořisti je konkurenčně vyloučena populací první). Tuto situaci znázorňuje obrázek 3.56.

Tomuto jevu, kdy populace přežívá ve společenstvu pouze za přítomnosti predátora, se v ekologii říká *koexistence zprostředkovaná predátorem* (predator mediated coexistence). Predátor je v tomto smyslu obligátním komensálem své kořisti.



Obrázek 3.56: Řešení modelu predátora živícího se dvěma konkurujícími si populacemi za nepřítomnosti predátora.

Predátor živící se dvěma nekonkurujícími si populacemi

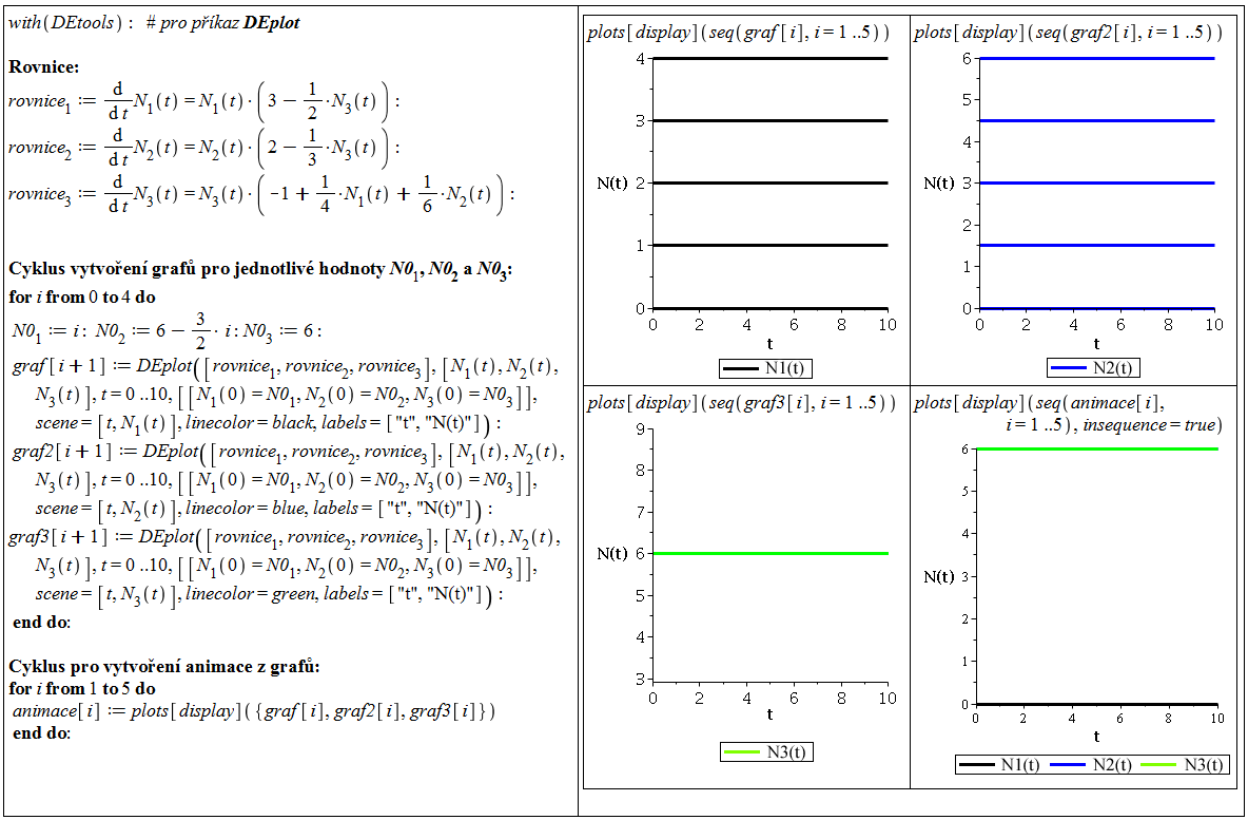
Model je popsán systém rovnic

$$\begin{aligned} N_1'(t) &= N_1(t)\cdot\big(\alpha_1 - \beta_{13}\cdot N_3(t)\big),\\ N_2'(t) &= N_2(t)\cdot\big(\alpha_2 - \beta_{23}\cdot N_3(t)\big),\\ N_3'(t) &= N_3(t)\cdot\big(-\alpha_3 + \beta_{31}\cdot N_1(t) + \beta_{32}\cdot N_2(t)\big). \end{aligned}$$

V tomto případě opět $N_1 = N_1(t)$ a $N_2 = N_2(t)$ označují velikosti populací kořisti, $N_3 = N_3(t)$ označuje velikost populace predátora. U populací kořisti nenastává mezidruhová ani vnitrodruhová konkurence; taková situace může nastat v případě, že zdroje pro populace kořistí jsou prakticky neomezené.

Při volbě parametrů

$$\alpha_1 = 3,\ \alpha_2 = 2, \alpha_3 = 1, \beta_{13} = \frac{1}{2},\ \beta_{23} = \frac{1}{3},\ \beta_{31} = \frac{1}{4},\ \beta_{32} = \frac{1}{6},$$



Obrázek 3.57: Řešení modelu predátora živícího se dvěma nekonkurujícími si populacemi pro $N_1(0) = p, N_2(0) = 6 - \frac{3}{2} \cdot p, N_3(0) = 6.$

a při počátečních hodnotách

$$N_1(0) = p,\ \ N_2(0) = 6 - \frac{3}{2} \cdot p,\ \ N_3(0) = 6,$$

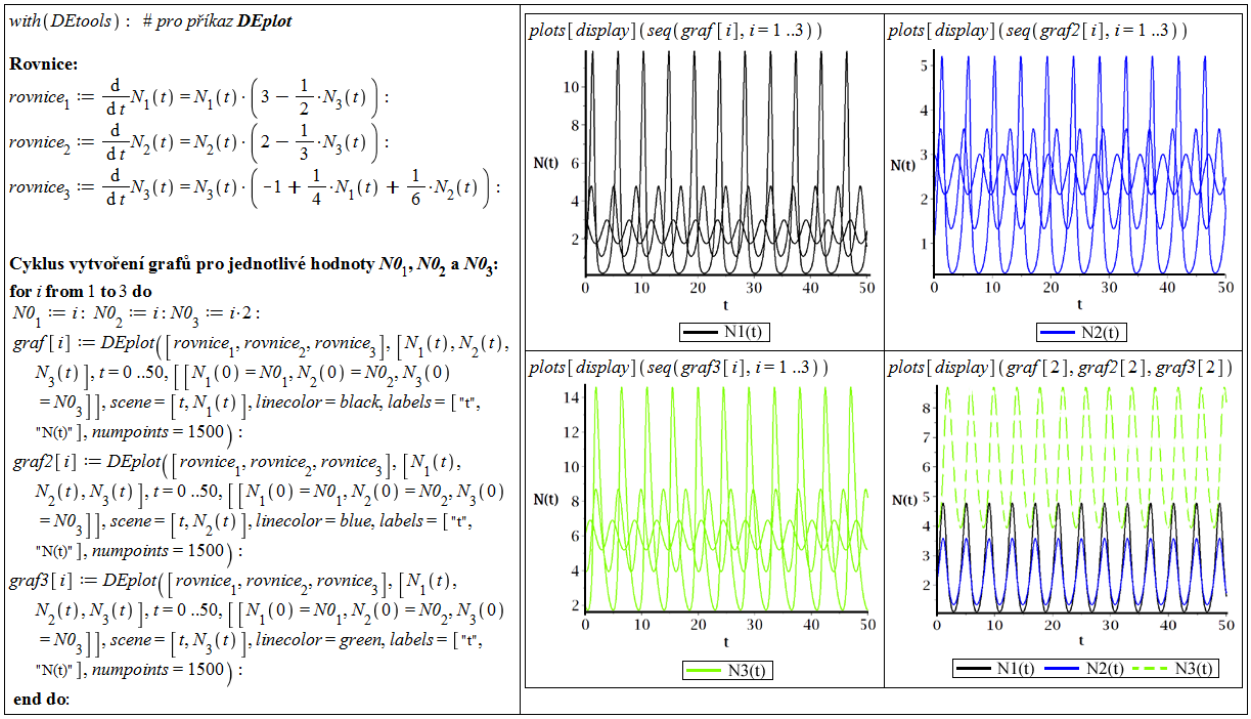
kde p je libovolné číslo z intervalu $[0, 4]$, velikosti všech populací zůstavají konstantní, jak dokumentuje obrázek 3.57. Poslední z grafů je tvořen animací vykreslující všechna tři řešení současně v závislosti na počátečních hodnotách.

Při stejných parametrech, ale jiných kladných počátečních hodnotách, všechny populace dlouhodobě přežívají a jejich velikosti kolísají. Tuto situaci znázorňuje obrázek 3.58. Jsou v něm zobrazena zvlášť jednotlivá řešení, poslední graf ukazuje všechna tři řešení současně pro počáteční velikosti populací: $N_1(0) = 2, N_2(0) = 2, N_3(0) = 4$.

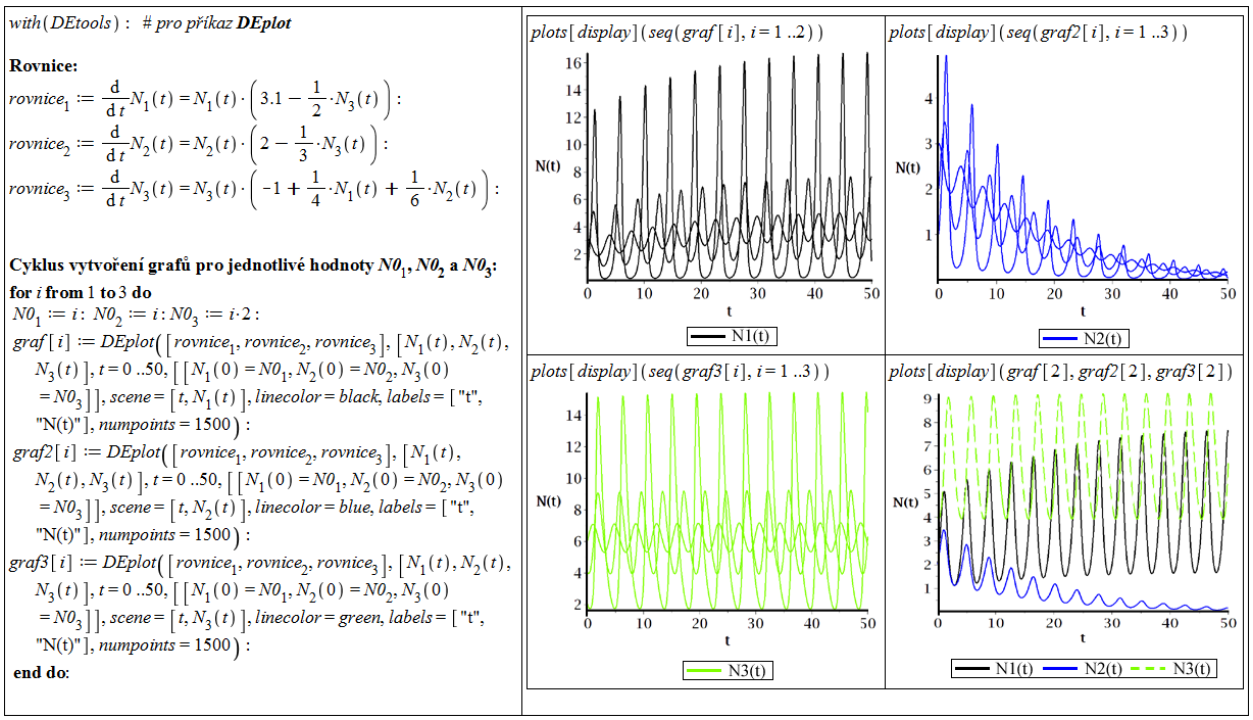
Pokud nepatrně změníme jeden z parametrů α_1, β_{13}, jedna z populací kořisti vymře. Při zvětšení α_1 nebo zmenšení β_{13} vymře druhá populace; při zmenšení α_1 nebo zvětšení β_{13} vymře první populace. Analogické tvrzení platí pro změny parametrů α_2, β_{23}. Na obrázku 3.59 můžeme pozorovat jednotlivá řešení v případě zvětšeného parametru α_1 ($\alpha_1 = 3.1$). Pro počáteční hodnoty $N_1(0) = 1, N_2(0) = 2, N_3(0) = 6$ jsou zobrazena všechna tři řešení současně.

V tomto případě dlouhodobě koexistují všechna tři možná „podspolečenstva“ dvou druhů, společenstvo tvořené třemi druhy pouze při speciální volbě parametrů. Komplexnější společenstvo tedy není tak ekologicky stabilní jako společenstva méně komplexní.

Uvedené tři modely společenstev naznačují, že vztah komplexity a stability společenstev není nijak jednoduchý. Někdy jsou komplexnější společenstva stabilnější než společenstva méně komplexní, jindy naopak. Je tedy nutné upřesnit nebo specifikovat „konvenční názor“



Obrázek 3.58: Řešení modelu predátora živícího se dvěma nekonkurujícími si populacemi – různé počáteční hodnoty.



Obrázek 3.59: Řešení modelu predátora živícího se dvěma nekonkurujícími si populacemi – různé počáteční hodnoty, jedna z populací vymírá.

ekologie, že větší složitost společenstva vede k jeho vyšší stabilitě, který vycházel z prací MacArthura[22] a Eltona [5] a byl zastávaný i v autoritativní Odumově učebnici [25]; podrobně je tato problematika diskutována v monografii [3, kapitola 23].

4 Modely se zahrnutím neurčitosti

V předchozí kapitole jsme si ukázali několik příkladů matematických modelů. Model je přitom vždy zjednodušením pozorovaného systému, a obsahuje tak prvky, kterým říkáme *neurčitosti*. V první kapitole jsme si neurčitosti ovlivňující model rozdělili do tří kategorií. Nyní se podíváme blíže na některé populační modely předchozí kapitoly z pohledu datové neurčitosti, respektive neurčitosti v parametrech modelu. Pro jednoduchost budeme vždy uvažovat pouze spojitý případ.

4.1 Analýza neurčitosti

Nejpoužívanějšími přístupy k reprezentaci neurčitostí v hodnotách parametrů modelu jsou [11], [14], [32]:

1. *Intervalová aritmetika* – užívá se k popisu datové neurčitosti vznikající buď nepřesností měření, nebo existencí několika alternativních metod (technik) k odhadu parametrů. Cílem intervalové analýzy je odhadnutí mezí výstupu modelu na základě mezí vstupů. V tomto přístupu se o vstupech nepředpokládá žádná další znalost, pouze ohraničení hodnot, jakých mohou nabývat. Každý neurčitý vstup je tak chápán jako interval, typicky $[x_i - \varepsilon, x_i + \varepsilon]$, kde x_i je „jakási“ střední hodnota příslušného parametru, který může nabývat libovolné hodnoty z uvedeného intervalu. Přitom nemáme k dispozici informaci o pravděpodobnostním rozložení těchto hodnot.

   Hlavní výhodou intervalové analýzy je schopnost popisovat neurčitosti bez znalosti pravděpodobnostní struktury vstupů. To je současně i nevýhodou, neboť charakterizace neurčitosti na výstupu je opět pouze prostřednictvím intervalu. V případech, kdy známe pravděpodobnostní rozložení hodnot vstupů, není tento typ analýzy doporučován [11], [26].

2. *Fuzzy teorie* – metoda umožňující provádět analýzu neurčitostí v případech, kdy jsou neurčitosti způsobeny vágností či nejasností. Používá se při zacházení s parametry, které jsou konstantní (nenáhodné), není možné je měřit, ale jejich hodnota je známa. Často se přitom jedná o nematematickou hodnotu (např. „asi dvě“, „tmavě zelený“ atp.). Fuzzy teorie je vhodnější ke kvalitativnímu posuzování než pro kvantitativní odhad neurčitosti [11], [14].

3. *Pravděpodobnostní analýza* – nejrozšířenější metoda popisu neurčitostí v modelech. Jednotlivé parametry jsou chápány jako náhodné veličiny s příslušným pravděpodobnostním rozložením hodnot, kterých mohou nabývat. Cílem pravděpodobnostní analýzy je určení pravděpodobnostního rozložení hodnot výstupu modelu. Pravděpodobnostní analýza má dva hlavní kroky. Prvním je určení pravděpodobnostního rozložení hodnot vstupních parametrů. To je zpravidla získáváno z odborné literatury, rozhodnutím experta, či z empirických dat. Druhým krokem analýzy je stanovení propagace neurčitosti (pravděpodobnostních rozložení) modelem.

4.2 Analýza citlivosti

S analýzou neurčitostí je úzce spojena tzv. *analýza citlivosti*. Zatímco při analýze neurčitostí charakterizujeme veškeré neurčitosti na vstupu a chceme získat celkovou neurčitost na výstupu modelu, při analýze citlivosti zkoumáme, jak konkrétně ovlivňují výstup jednotlivé parametry modelu. Metody analýzy citlivosti rozdělujeme na *lokální* a *globální*. Lokální metody určují, jak citlivý je výstup modelu na změny vstupních parametrů v „těsné blízkosti“ nějaké jejich reprezentativní hodnoty (typicky střední hodnoty intervalu nejistoty). V modelech, kde některý z parametrů nemá žádnou standardní hodnotu (a může nabývat širšího spektra hodnot), využíváme globálních metod analýzy citlivosti. Výsledky analýzy napoví, které parametry je důležité kontrolovat (a znát tak co nejpřesněji jejich hodnoty) [6], [32].

V tomto textu se budeme zabývat pouze analýzou lokální citlivosti, čtenáře se zájmem o informace k analýze globální citlivosti odkážeme například na [29].

Předpokládejme pro jednoduchost model popsaný rovnicí

$$\frac{dy(t,\boldsymbol{x})}{dt} = f(y(t,\boldsymbol{x})), \tag{4.1}$$

kde $y(t,\boldsymbol{x})$ je hledaná funkce (řešení), t čas, $\boldsymbol{x} = (x_1, x_2, ..., x_m)^T$ je m-rozměrný vektor parametrů modelu, f funkce v proměnné $y(t,\boldsymbol{x})$. Efekt změny parametrů je možné popsat Taylorovým rozvojem

$$y(t,\boldsymbol{x}+\Delta\boldsymbol{x}) = y(t,\boldsymbol{x}) + \sum_{j=1}^{m} \frac{\partial y(t,\boldsymbol{x})}{\partial x_j}\cdot\Delta x_j + \sum_{i=1}^{m}\sum_{j=1}^{m} \frac{\partial^2 y(t,\boldsymbol{x})}{\partial x_i \partial x_j}\cdot\Delta x_i\cdot\Delta x_j + \cdots, \tag{4.2}$$

kde $\Delta\boldsymbol{x} = (\Delta x_1, \Delta x_2, ..., \Delta x_m)^T$. Parciální derivace $\frac{\partial y(t,\boldsymbol{x})}{\partial x_j}$ se nazývají *lokální citlivosti prvního řádu*, výrazy $\frac{\partial^2 y(t,\boldsymbol{x})}{\partial x_i \partial x_j}$ se nazývají *lokální citlivosti druhého řádu* atd. Ne vždy jsme schopni určit lokální citlivosti analyticky. Pro jejich výpočet proto existuje několik numerických metod, z nichž zmíníme metodu konečných diferencí.

Metoda konečných diferencí

Nejjednodušší způsob výpočtu lokální citlivosti je založen na nahrazení derivace její aproximací:

$$\frac{\partial y(t,\boldsymbol{x})}{\partial x_j} \approx \frac{y(t, x_1, ..., x_j+\Delta x_j, ..., x_m) - y(t,\boldsymbol{x})}{\Delta x_j}, \qquad j = 1, ..., m. \tag{4.3}$$

Tato procedura bývá též označována jako *metoda hrubé síly* nebo *nepřímá metoda* [28].

4.3 Neomezený růst populace živých organismů

První model, který jsme vytvořili, představoval neomezený růst populace. Popisovala jej rovnice (3.5)

$$N'(t) = (a - b) \cdot N(t)$$

s počáteční podmínkou

$$N(0) = N0. \tag{4.4}$$

Obecné řešení rovnice je tvaru:

$$N(t) = N0 \cdot e^{(a-b)\cdot t}. \tag{4.5}$$

Pro zobrazení konkrétního řešení jsme uvažovali populaci zajíce polního, o němž jsme z dostupných zdrojů [23] zjistili údaje o porodnosti a věku dožití. Získané informace v sobě přitom zahrnovaly neurčitost (viz str. 25):

věk dožití: 10-12 let
porodnost: 3-4 krát ročně 1-7 mláďat (tj. 3-28 mláďat ročně)

Pro koeficienty porodnosti a úmrtnosti v případě „měsíčního kroku“ ve skutečnosti platí:[28]

$$a \in [0.125, 1.17], \quad b \in [0.007, 0.0083].$$

4.3.1 Analýza neurčitosti

Parametr $N0$ jsme zvolili sami „náhodně“. Budeme tedy uvažovat, že jeho hodnota je přesná a do analýzy neurčitosti jej zahrnovat nebudeme. Vzhledem k povaze informací o parametrech se jeví jako nejvhodnější analýza intervalovou aritmetikou.

Intervalová aritmetika

Práci s intervalovou aritmetikou v systému Maple umožňuje balík **Tolerances**. S jeho pomocí definujeme parametry a, b jako intervaly. Řešení $N(t)$ pak bude „intervalová“ funkce, tj. funkce, jejíž funkční hodnotu pro libovolné t bude tvořit interval hodnot. Zápis intervalové funkce $N(t)$ v Maple může být nepřehledný a matoucí. Při práci s balíkem **Tolerances** proto musíme být obezřetní, viz obrázek 4.1.

Intervalovou funkci $N(t)$ nemůžeme vykreslit jako „klasickou“ funkci. Přesto máme několik možností k jejímu zobrazení. Jednak můžeme funkci „rozdělit“ na více klasických funkcí (např. nejmenší, střední, největší, ...) a ty vykreslit do jednoho grafu, abychom znázornili průběh funkce pro nějaké významné hodnoty parametrů z jejich intervalů neurčitosti. Například obrázek 4.2 ilustruje funkci $N(t)$ pro nejnižší a nejvyšší hodnotu rozdílu $a-b$ (čárkované křivky) a střední hodnotu rozdílu $a - b$ (plná křivka).

Další možnost je vykreslit funkci trojrozměrně jako funkci dvou proměnných. První proměnnou bude čas t, druhou proměnnou bude parametr, jehož vliv na řešení chceme pozorovat. Na obrázku 4.3 je jako druhý parametr zvolen rozdíl $a-b$, který označíme r, jak jsme zavedli již dříve.

[28] Na str. 22 jsme uvažovali koeficient porodnosti z intervalu [0,1.17] (snížili jsme tedy dolní mez až na 0), abychom mohli při vykreslování řešení pozorovat i situaci, kdy $a \leq b$. Tato situace pro získané údaje o zajíci polním nenastane, ale z obecného pohledu, jak se řešení modelu může chovat, je podstatná.

Rovnice modelu a její řešení: $\frac{d}{dt}N(t) = (a-b)\cdot N(t), N(0) = N0 \xrightarrow{\text{solve DE}} N(t) = N0\, e^{(a-b)\,t}$

Načtení balíku pro práci s intervalovou aritmetikou: $with(Tolerances)$:

Nastavení parametrů: $a := 0.6475 \,\&+-\, 0.5225;$ $b := 0.00765 \,\&+-\, 0.00065;$ $N0 := 100;$

$$0.6475 \pm 0.5225$$
$$0.00765 \pm 0.00065$$
$$100$$

Definice funkce řešení: $N(t) := N0\cdot e^{(a-b)\cdot t}$

$$t \to N0\, e^{(a+Tolerances:-`-`(b))\,t}$$

Řešení v různých časových bodech:
$N(1) = 216. \pm 104.$
$N(5) = 16900. \pm 16700.$
$N(10) = 5.62\,10^6 \pm 5.62\,10^6$

Střední hodnota intervalu neurčitosti v řešení:
$NominalValue(N(1)) = 216.1649844$
$NominalValue(N(5)) = 16854.18495$
$NominalValue(N(10)) = 5.621176712\,10^6$

Polovina intervalu neurčitosti v řešení:
$ToleranceValue(N(1)) = 103.7867600$
$ToleranceValue(N(5)) = 16674.95490$
$ToleranceValue(N(10)) = 5.620855480\,10^6$

Obrázek 4.1: Řešení neomezeného růstu populace s balíkem Tolerances.

Rovnice modelu a její řešení: $\frac{d}{dt}N(t) = (a-b)\cdot N(t), N(0) = N0 \xrightarrow{\text{solve DE}} N(t) = N0\, e^{(a-b)\,t}$

Načtení balíku pro práci s intervalovou aritmetikou: $with(Tolerances)$:

Nastavení parametrů: $a := 0.6475 \,\&+-\, 0.5225$: $b := 0.00765 \,\&+-\, 0.00065$: $N0 := 100$:

Nejnižší hodnota rozdílu *a-b*: $NominalValue(a-b) - ToleranceValue(a-b) = 0.1167000000$

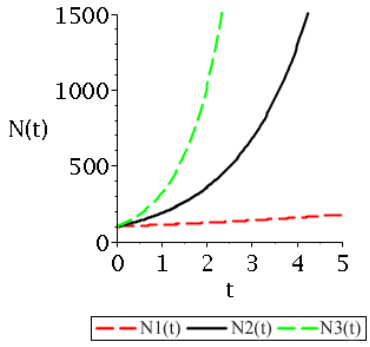
Střední hodnota rozdílu *a-b*: $NominalValue(a-b) = 0.6398500000$

Nejvyšší hodnota rozdílu *a-b*: $NominalValue(a-b) + ToleranceValue(a-b) = 1.163000000$

Definice jednotlivých funkcí: $N_1(t) := N0\cdot e^{0.1167\cdot t}$: $N_2(t) := N0\cdot e^{0.63985\cdot t}$: $N_3(t) := N0\cdot e^{1.163\cdot t}$:

Vykreslení řešení:

$plot([N_1(t), N_2(t), N_3(t)], t = 0..5, y = 0..1500, thickness = 2, color = [red, black, green], linestyle = [dash, solid, dash], legend = ["N1(t)","N2(t)","N3(t)"], labels = ["t", "N(t)"], font = [Times, roman, 15], labelfont = [Times, roman, 15])$



Obrázek 4.2: Vykreslení jednotlivých řešení z intervalu neurčitosti.

Rovnice modelu a její řešení: $\frac{d}{dt}N(t) = (a-b)\cdot N(t), N(0) = N0 \xrightarrow{\text{solve DE}} N(t) = N0\, e^{(a-b)\,t}$

Načtení balíku pro práci s intervalovou aritmetikou: $with(Tolerances)$:

Nastavení parametrů: $a := 0.6475 \,\&+- 0.5225$: $b := 0.00765 \,\&+- 0.00065$: $N0 := 100$:

Rozdíl *a-b*:

$a - b = [NominalValue(a-b) - ToleranceValue(a-b), NominalValue(a-b) + ToleranceValue(a-b)]$

$$0.640 \pm 0.523 = [0.1167000000, 1.163000000]$$

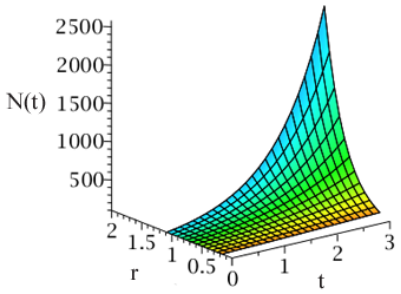
Definice řešení jako funkce dvou parametrů:

$N(t,r) := piecewise(0.1167 < r < 1.163, N0\cdot e^{r\cdot t}, undefined)$

$$(t,r) \rightarrow piecewise(0.1167 < r \text{ and } r < 1.163, N0\, e^{r\,t}, undefined)$$

Vykreslení řešení:

$plot3d(N(t,r), t = 0..3, r = 0..2, axes = frame, color = r, font = [Times, roman, 15], labelfont = [Times, bold, 15], labels = ["t", "r", "N(t)"])$



Obrázek 4.3: Vykreslení řešení jako funkce dvou parametrů.

Fuzzy teorie

Fuzzy množiny jsou rozšířením klasických množin. Fuzzy množina modeluje přibližné neurčité hodnoty, které mohou být čísla i nominální hodnoty (např. „skoro zelený, mírně do modra“). Zatímco u klasických množin prvek do množiny buď patří, či nikoliv, u fuzzy množin může prvek náležet do množiny jen částečně. Klasickou množinu jednoznačně určuje její charakteristická funkce, která nabývá hodnot 0 nebo 1. Naproti tomu fuzzy množinu popisuje tzv. *funkce příslušnosti* μ nabývající libovolné hodnoty z uzavřeného intervalu $[0, 1]$. Pro počítání s fuzzy množinami představujícími číselné hodnoty existuje několik typů fuzzy aritmetiky, například fuzzy obdoba intervalové aritmetiky, kdy se intervalová aritmetika aplikuje na intervaly na hladinách $\mu = 0$ a $\mu = 1$. Tento typ aritmetiky využijeme i v našem případě [12].

Ve chvíli, kdy obdržíme o parametrech modelu nějakou další informaci, například: průměrné plození nových jedinců je nejběžnější, zatímco mezní případy (3, resp. 28, mláďat ročně) se téměř nevyskytují, můžeme koeficient porodnosti chápat jako fuzzy množinu. V tomto případě bychom mohli koeficientu porodnosti přiřadit fuzzy množinu, jejíž funkce příslušnosti má trojúhelníkový tvar s vrcholem v bodě průměrného plození. Podobnou úvahou (např. že věk dožití je nejčastěji 10-11 let, 12 let pouze výjimečně) bychom přiřadili koeficientu úmrtnosti fuzzy množinu, jejíž funkce příslušnosti má tvar pravoúhlého lichoběžníku (v němž není přítomná svislá úsečka, aby se jednalo o zobrazení).

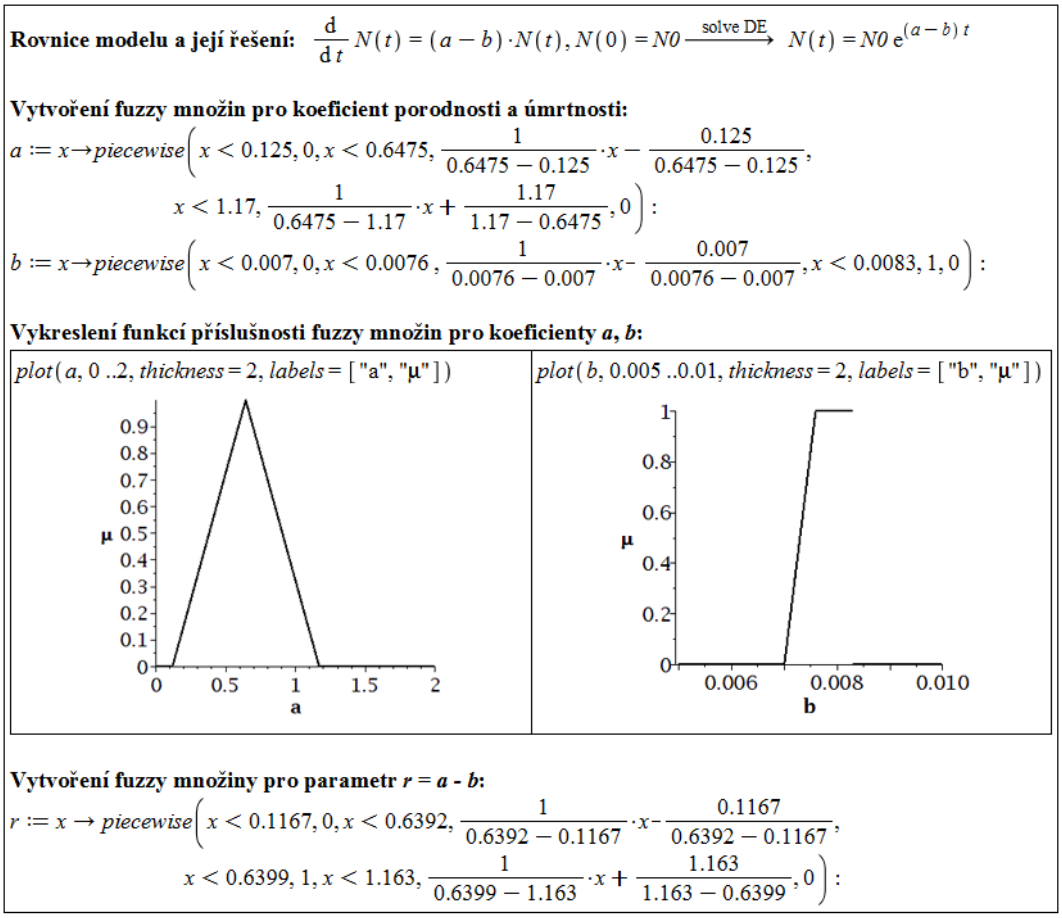
Zmíněné úvahy nyní provedeme. Práci s fuzzy množinami v systému Maple umožňuje toolbox **FuzzySets**. V našem případě vystačíme bez něj, přičemž funkce příslušnosti budeme definovat po částech pomocí příkazu **piecewise**. Na obrázku 4.4 tak definujeme funkce

příslušnosti pro fuzzy množiny koeficientu porodnosti a a koeficientu úmrtnosti b. Jelikož se ve skutečnosti jedná o fuzzy čísla, která potřebujeme odečíst (určit rozdíl $a - b$), musíme vytvořit funkci příslušnosti r pro fuzzy množinu rozdílu $a - b$. Na tomto místě využijeme intervalovou aritmetiku k určení bodů na hladinách $\mu = 0$ a $\mu = 1$, v nichž dochází ke změně směrnice funkce μ. Na hladině $\mu = 0$ tak získáme dva body reprezentující nejnižší a nejvyšší možnou hodnotu rozdílu $a - b$:

$$0.125 - 0.0083 = 0.1167, \quad 1.17 - 0.007 = 1.163,$$

na hladině $\mu = 1$ analogicky body

$$0.6475 - 0.0083 = 0.6392, \quad 0.6475 - 0.0076 = 0.6399.$$



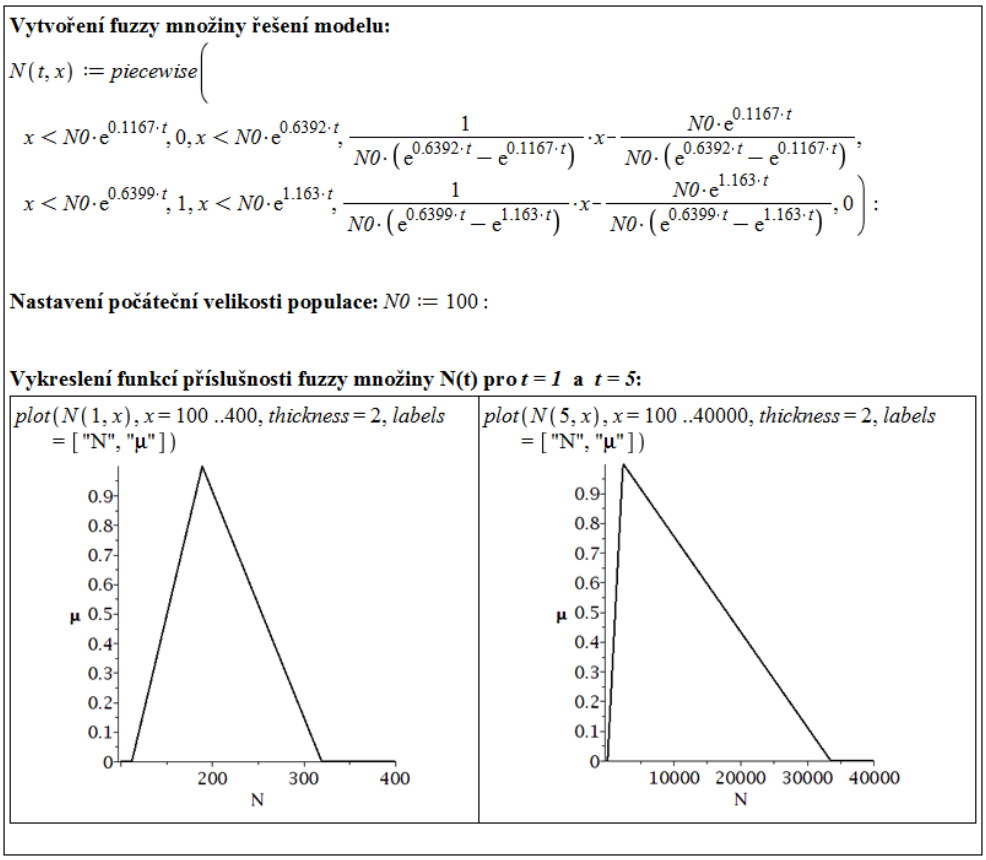
Obrázek 4.4: Definice koeficientů porodnosti a úmrtnosti jako fuzzy množin.

Díky analytickému předpisu (4.5), řešení modelu, můžeme vytvořit také funkci příslušnosti pro fuzzy množinu funkce $N(t)$. Tato funkce příslušnosti je časově závislá, přičemž má stále stejný lichoběžníkový tvar (pro různá t) a body, v nichž se mění směrnice funkce příslušnosti, vychází z bodů právě určených pro fuzzy množinu r. Získáme tak body

$$N0 \cdot e^{(0.125-0.0083)\cdot t}, \; N0 \cdot e^{(0.6475-0.0083)\cdot t}, \; N0 \cdot e^{(0.6475-0.0076)\cdot t}, \; N0 \cdot e^{(1.17-0.007)\cdot t}.$$

Na obrázku 4.5 je definována fuzzy množina $N(t)$ jako funkce dvou proměnných. Pro časové body $t = 1$ a $t = 5$ je následně vykreslena. Velmi krátký interval na hladině $\mu = 1$ způsobuje, že funkce příslušnosti má na pohled spíše trojúhelníkový tvar.

Získaná fuzzy množina pro $N(t)$ nám říká, jak může řešení ve zvoleném čase vypadat, tedy kolik jedinců populace v tomto čase může mít. Oproti intervalové aritmetice je každý údaj podpořen hodnotou příslušnosti, tedy mírou možnosti.



Obrázek 4.5: Řešení jako fuzzy množina.

Pravděpodobnostní analýza

Při pravděpodobnostní analýze chápeme jednotlivé parametry modelu jako náhodné veličiny, kterým přiřadíme pravděpodobnostní rozložení hodnot, jichž mohou nabývat. V našem případě budeme postupovat podobně jako v případě užití fuzzy množin. Za náhodné veličiny budeme považovat koeficienty porodnosti a úmrtnosti. Budeme přitom opět vycházet z údajů o zajíci polním, které jsme získali z literatury. Koeficientu porodnosti přiřadíme trojúhelníkové rozložení hodnot na intervalu $[0.125, 1.17]$ s maximem v bodě 0.6475, který odpovídá průměrnému plození. Pro koeficient úmrtnosti nemáme v systému Maple k dispozici lichoběžníkové rozložení. Máme dvě možnosti, jak to vyřešit. Buď rozložení vytvoříme, přičemž jeho parametry určíme z toho, že integrál z hustoty rozložení (tj. obsah plochy mezi funkcí pravděpodobnostního rozložení a osou x) je roven 1. Nebo použijeme nějaké existující rozložení nejvíc „podobné“ tomu, které chceme. V našem případě bychom pro koeficient úmrtnosti zvolili například trojúhelníkové rozložení hodnot na intervalu $[0.007, 0.0083]$ s maximem v bodě 0.0076, který odpovídá průměrné úmrtnosti.

Druhým krokem pravděpodobnostní analýzy je stanovení šíření neurčitosti. To provádíme generováním „dostatečného“ množství n hodnot náhodných veličin a následným vyhodnocováním modelu. Získáme tak pravděpodobnostní rozložení hodnot řešení modelu. „Dostatečné“ n se pohybuje v řádech tisíců až milionů, jeho hodnota závisí na více faktorech. Při

Rovnice modelu a její řešení: $\frac{\mathrm{d}}{\mathrm{d}t} N(t) = (a-b)\cdot N(t), N(0) = N0 \xrightarrow{\text{solve DE}} N(t) = N0\, \mathrm{e}^{(a-b)\,t}$

Načtení balíku *Statistics*: *with*(*Statistics*) :

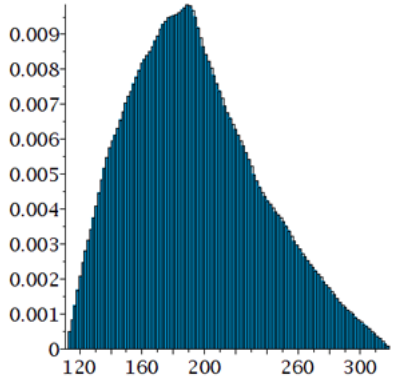
Vytvoření náhodných veličin pro koeficient porodnosti a úmrtnosti:
A := *RandomVariable*(*Triangular*(0.125, 1.17, 0.6475)) : *B* := *RandomVariable*(*Triangular*(0.007, 0.0083, 0.0076)) :

Vygenerování hodnot, nastavení počáteční velikosti populace:
koef_a := *Sample*(*A*, 10^5) : *koef_b* := *Sample*(*B*, 10^5) : *N0* := 100 :

Výpočet řešení N(t) a jeho vykreslení pro *t* = *1* a *t* = *5*:

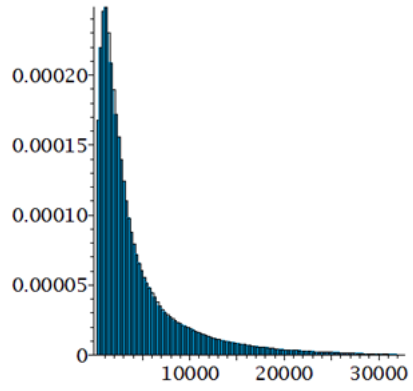
reseni_1 := *Vector*(10^5) :
for *i* **from** 1 **to** 10^5 **do**
reseni_1[*i*] := N0·$\mathrm{e}^{(koef\_a[i]-koef\_b[i])}$
end do:

Histogram(*reseni_1*, *averageshifted* = 4)



reseni_5 := *Vector*(10^5) :
for *i* **from** 1 **to** 10^5 **do**
reseni_5[*i*] := N0·$\mathrm{e}^{(koef\_a[i]-koef\_b[i])\cdot 5}$
end do:

Histogram(*reseni_5*, *averageshifted* = 4)



Obrázek 4.6: Pravděpodobnostní analýza neurčitostí.

prováděnípravděpodobnostní analýzy se zpravidla postupuje tak, že se zvolí n na základě zkušeností a možností (případně potřeb) a vygenerují se dvě skupiny po n vygenerovaných hodnotách. Pokud se shodují základní statistické parametry (střední hodnota, rozptyl) obou skupin, je n považováno za dostatečné. Pokud se statistické parametry liší, je nutné zvýšit počet generovaných hodnot.

Obrázek 4.6 ilustruje provedení pravděpodobnostní analýzy v Maple. Nejprve vytvoříme náhodné veličiny koeficientů porodnosti a úmrtnosti (které označíme velkými písmeny). Pro oba koeficienty použijeme již implementované trojúhelníkové rozložení pravděpodobnosti. Příkazem **Sample** vygenerujeme 10^5 hodnot každé z náhodných veličin a 10^5krát pro ně vyhodnotíme řešení modelu v případě $t = 1$ a v případě $t = 5$. Příkazem **Histogram** zobrazíme histogram získaných řešení (parametrem **averageshifted** zvýšíme počet vykreslených sloupců). Z výsledků můžeme pozorovat, jak se v čase mění pravděpodobnostní rozložení hodnot řešení. V čase $t = 1$ můžeme ještě mluvit o přibližně trojúhelníkovém, v čase $t = 5$ se histogram podobá spíše log-normálnímu rozložení hodnot.

4.3.2 Analýza citlivosti

Předcházející analýzou neurčitosti jsme kvantifikovali celkovou neurčitost v řešení modelu. Nyní zjistíme, jak je řešení citlivé na hodnoty jednotlivých parametrů. Jelikož máme

analytický předpis řešení (4.5)

$$N(t) = N0 \cdot e^{(a-b)\cdot t},$$

můžeme určit lokální citlivosti parametrů přímo z definice (4.2). Nás zajímá především velikost citlivosti, takže budeme lokální citlivosti chápat jako absolutní hodnoty z definovaných výrazů. Označme citlivosti písmenem s, index nechť značí parametr, jehož citlivost uvažujeme. Potom:

$$s_a = \left|\frac{\partial N(t,a,b,N0)}{\partial a}\right| = \left|\frac{\partial N0 \cdot e^{(a-b)\cdot t}}{\partial a}\right| = N0 \cdot t \cdot e^{(a-b)\cdot t},$$

$$s_b = \left|\frac{\partial N(t,a,b,N0)}{\partial b}\right| = \left|\frac{\partial N0 \cdot e^{(a-b)\cdot t}}{\partial b}\right| = N0 \cdot t \cdot e^{(a-b)\cdot t},$$

$$s_{N0} = \left|\frac{\partial N(t,a,b,N0)}{\partial N0}\right| = \left|\frac{\partial N0 \cdot e^{(a-b)\cdot t}}{\partial N0}\right| = e^{(a-b)\cdot t}.$$

Lokální citlivosti jsou časově závislé. Citlivosti koeficientů úmrtnosti a porodnosti jsou stejné, citlivost počáteční velikosti populace je pro $t > \frac{1}{N0}$ nižší než citlivosti zbylých parametrů. Chování modelu tedy ovlivňují nejvíce koeficient porodnosti a koeficient úmrtnosti.

4.4 Omezený růst populace

U následujících modelů provedeme pouze analýzu citlivosti, neboť k analýze neurčitosti již nemáme dostatečné údaje. Uvažujme nyní modifikovanou variantu předchozího modelu, kterou jsme odvodili v sekci 3.1.4.

4.4.1 Analýza citlivosti

V případě omezeného růstu populace jsme dospěli k řešení

$$N(t) = \frac{N0 \cdot (a-b)}{N0 \cdot c - (N0 \cdot c - a + b) \cdot e^{-(a-b)\cdot t}}.$$

Pomocí systému Maple můžeme jednoduše získat citlivosti jako parciální derivace funkce $N(t,a,b,c,N0)$ závislé na parametrech, výsledné výrazy však budou stěží interpretovatelné, jak můžeme vidět na obrázku 4.7. Vyhodnoťme proto citlivosti pro „průměrné“ nastavení parametrů, které jsme použili již dříve: $a = \frac{7}{12}, b = \frac{1}{11\cdot 12}, c = \frac{1}{11\cdot 12}, N0 = 100$, a pro $t = 10$ (obrázek 4.8). Vidíme, že v tomto případě je nejcitlivější parametr c, tedy lineární člen koeficientu úmrtnosti. Počáteční velikost populace je opět nejméně citlivý parametr, což odpovídá chování modelu (neboť řešení se ustálí na stejné hodnotě pro libovolnou počáteční velikost populace).

Výpočet parciální derivace:

$$\frac{\partial}{\partial a}\frac{N0\,(a-b)}{N0\cdot c-(N0\,c-a+b)\,e^{-(a-b)\,t}}$$

$$\frac{N0}{N0\,c-(N0\,c-a+b)\,e^{-(a-b)\,t}}-\frac{N0\,(a-b)\left(e^{-(a-b)\,t}+(N0\,c-a+b)\,t\,e^{-(a-b)\,t}\right)}{\left(N0\,c-(N0\,c-a+b)\,e^{-(a-b)\,t}\right)^2}$$

Úprava získaného výsledku:

simplify(%)

$$-\left(N0\left(-N0\,c+e^{-(a-b)\,t}N0\,c+e^{-(a-b)\,t}a\,t\,N0\,c-e^{-(a-b)\,t}t\,a^2+2\,e^{-(a-b)\,t}a\,t\,b\right.\right.$$
$$\left.\left.-e^{-(a-b)\,t}b\,t\,N0\,c-e^{-(a-b)\,t}t\,b^2\right)\right)\Big/\left(-N0\,c+e^{-(a-b)\,t}N0\,c-e^{-(a-b)\,t}a+e^{-(a-b)\,t}b\right)^2$$

Další zjednodušující úprava získaného výrazu:

collect$\left(\%,\,e^{-(a-b)\,t}\right)$

$$-\frac{N0\left(\left(N0\,c+a\,t\,N0\,c-t\,a^2+2\,a\,t\,b-b\,t\,N0\,c-t\,b^2\right)e^{-(a-b)\,t}-N0\,c\right)}{\left((N0\,c-a+b)\,e^{-(a-b)\,t}-N0\,c\right)^2}$$

Obrázek 4.7: Výpočet obecného předpisu pro citlivost koeficientu porodnosti v případě omezeného růstu populace.

Citlivost koeficientu porodnosti *a*:

$$s_a:=\left|eval\left(\frac{\partial}{\partial a}\frac{N0\,(a-b)}{N0\cdot c-(N0\,c-a+b)\,e^{-(a-b)\,t}},\left\{a=\frac{7}{12},b=\frac{1}{11\cdot 12},c=\frac{1}{11\cdot 12},N0=100,t=10\right\}\right)\right|:$$

$evalf(s_a)$

131.2057452

Citlivost konstantního členu z koeficientu úmrtnosti *b*:

$$s_b:=\left|eval\left(\frac{\partial}{\partial b}\frac{N0\,(a-b)}{N0\cdot c-(N0\,c-a+b)\,e^{-(a-b)\,t}},\left\{a=\frac{7}{12},b=\frac{1}{11\cdot 12},c=\frac{1}{11\cdot 12},N0=100,t=10\right\}\right)\right|:$$

$evalf(s_b)$

131.2057452

Citlivost lineárního členu z koeficientu úmrtnosti c:

$$s_c:=\left|eval\left(\frac{\partial}{\partial c}\frac{N0\,(a-b)}{N0\cdot c-(N0\,c-a+b)\,e^{-(a-b)\,t}},\left\{a=\frac{7}{12},b=\frac{1}{11\cdot 12},c=\frac{1}{11\cdot 12},N0=100,t=10\right\}\right)\right|:$$

$evalf(s_c)$

10015.49110

Citlivost počáteční velikosti populace *N0*:

$$s_{N0}:=\left|eval\left(\frac{\partial}{\partial N0}\frac{N0\,(a-b)}{N0\cdot c-(N0\,c-a+b)\,e^{-(a-b)\,t}},\left\{a=\frac{7}{12},b=\frac{1}{11\cdot 12},c=\frac{1}{11\cdot 12},N0=100,t=10\right\}\right)\right|:$$

$evalf(s_{N0})$

0.0018272692

Obrázek 4.8: Výpočet citlivostí jednotlivých parametrů v případě omezeného růstu populace.

4.5 Populace pod tlakem nespecializovaného predátora

4.5.1 Analýza citlivosti

Poslední model, u nějž provedeme analýzu citlivosti, je model růstu populace pod tlakem nespecializovaného predátora, viz sekce 3.2. Při analýze neomezeného růstu populace jsme vystačili s obecným analytickým předpisem citlivostí, u omezeného růstu populace jsme pro vyvození závěrů museli určit citlivosti pro konkrétní numerické nastavení parametrů, stále jsme však citlivosti počítali jako parciální derivace funkce řešení modelu.

V tomto případě již nemáme analytický předpis řešení modelu, musíme proto počítat citlivosti numericky (přibližně), což provedeme metodou konečných diferencí. Podle (4.3) tak pro citlivost parametru x_i získáme následující předpis:

$$s_{x_i} = \frac{\partial y(t, \boldsymbol{x})}{\partial x_i} \approx \frac{y(t, x_1, ..., x_{i-1}, x_i + \Delta x_i, x_{i+1}, ..., x_m) - y(t, \boldsymbol{x})}{\Delta x_i}. \tag{4.6}$$

Jak velkou diferenci k výpočtu použít závisí na modelu a jeho parametrech. Obvykle se volí diference 1 % nebo 10 % z hodnoty příslušného parametru. Abychom zamezili vlivu velikosti absolutní hodnoty uvažovaného parametru[29], vynásobíme předpis (4.6) jeho velikostí (tj. x_i). Zcela analogicky (abychom zamezili vlivu velikosti absolutní hodnoty řešení) naopak předpis (4.6) podělíme touto hodnotou (tj. $y(t, \boldsymbol{x})$). Celkově tak počítáme tzv. normalizovanou citlivost modelu $y(t, \boldsymbol{x})$ na parametr x_i danou předpisem:

$$s_{x_i} = \frac{\partial y(t, \boldsymbol{x})}{\partial x_i} \approx \frac{y(t, x_1, ..., x_{i-1}, x_i + \Delta x_i, x_{i+1}, ..., x_m) - y(t, \boldsymbol{x})}{\Delta x_i} \cdot \frac{x_i}{y(t, \boldsymbol{x})}. \tag{4.7}$$

Může se stát, že přičtení určité diference k hodnotě parametru způsobí jinou (velikostní) změnu řešení modelu než odečtení stejné diference od hodnoty parametru. Zpravidla se proto určí změna způsobená přičtením diference $(y(t, x_1, ..., x_{i-1}, x_i + \Delta x_i, x_{i+1}, ..., x_m) - y(t, \boldsymbol{x}))$ i jejím odečtením $(y(t, x_1, ..., x_{i-1}, x_i - \Delta x_i, x_{i+1}, ..., x_m) - y(t, \boldsymbol{x}))$ a do vztahu pro citlivost se vezme průměr těchto hodnot. Stejně jako dříve nás bude zajímat velikost citlivosti, takže budeme navíc uvažovat absolutní hodnoty ze zmíněných výrazů. Získáme tedy:[30]

$$s_{x_i} \approx \frac{E(|y(t, x_1, ..., x_{i-1}, x_i \pm \Delta x_i, x_{i+1}, ..., x_m) - y(t, \boldsymbol{x})|)}{\Delta x_i} \cdot \frac{x_i}{y(t, \boldsymbol{x})}. \tag{4.8}$$

Metodu konečných diferencí provedeme pro obě zmíněné diference (1 % a 10 %), příslušné citlivosti rozlišíme dalším indexem. V případě $\Delta x_i = 0.01 \cdot x_i$ budeme index citlivosti definovat předpisem

$$s_{x_i,1\%} = \frac{E(|y(t, x_1, ..., x_{i-1}, x_i \pm \Delta_1 x_i, x_{i+1}, ..., x_m) - y(t, \boldsymbol{x})|)}{y(t, \boldsymbol{x})} \cdot 100, \tag{4.9}$$

neboť $\frac{x_i}{\Delta x_i} = 100$. Analogicky pro $\Delta x_i = 0.1 \cdot x_i$ budeme index citlivosti definovat předpisem

$$s_{x_i,10\%} = \frac{E(|y(t, x_1, ..., x_{i-1}, x_i \pm \Delta_{10} x_i, x_{i+1}, ..., x_m) - y(t, \boldsymbol{x})|)}{y(t, \boldsymbol{x})} \cdot 10. \tag{4.10}$$

[29] tento vliv je způsobený výrazem Δx_i ve jmenovateli předpisu (4.6)

[30] $E(y)$ značí střední hodnotu (v tomto případě průměr) z možných hodnot y, tj. např. $E(f(x_i \pm \Delta x_i)) = \frac{1}{2} \cdot (f(x_i + \Delta x_i) + f(x_i - \Delta x_i))$

Model růstu populace pod tlakem nespecializovaného predátora je popsán rovnicí

$$N'(t) \;=\; r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) \;-\; \begin{cases} \dfrac{S}{N_{\text{krit}}} \cdot N(t) & \ldots \quad N(t) \le N_{\text{krit}}, \\ S & \ldots \quad N(t) > N_{\text{krit}}. \end{cases}, \quad N(0) = N0.$$

Nyní vypočítáme lokální citlivosti parametrů v případě, že

$$S < \frac{1}{4} \cdot r \cdot K, \quad N_{\text{krit}} \le \frac{S}{r}.$$

K tomu využijeme stejné nastavení parametrů, jaké jsme zvolili při analýze řešení na obrázku 3.24:

$$S = 200,\ r = 1,\ K = 1000,\ N_{\text{krit}} = 150$$

Počáteční hodnoty zvolíme tři různé:

$$N0_1 = 100,\ N0_2 = \frac{K}{2} \cdot \left(1 - \sqrt{1 - \frac{4 \cdot S}{r \cdot K}}\right) \doteq 276,\ N0_3 = 700.$$

Citlivosti přitom budeme počítat pro čas $t = 15$, v němž má sledovaná populace přibližně ustálenou velikost. Z prostorových důvodů ukazujeme pouze výpočet citlivosti parametru r (vnitřního koeficientu růstu populace) pro $N(0) = N0_1 = 100$ (obrázek 4.9). Při výpočtech citlivostí parametrů r, S a K musíme mít na paměti, že hodnota $N0_2$ je pomocí nich definována, tj. při změně kteréhokoli ze zmíněných parametrů musíme určit též odpovídající hodnotu $N0_2$.

Pro výpočet řešení modelu použijeme příkaz **dsolve** s parametrem **numeric** pro numerické řešení. Výsledek pro čas t zadaný do kulatých závorek získáme ve tvaru $[t = ..., N(t) = ...]$, v němž jsou místo teček numerické hodnoty. Jestliže z něj chceme získat pouze numerickou hodnotu funkce $N(t)$, musíme do hranatých závorek za příkaz specifikovat, že chceme druhý údaj. Příkazem **rhs** z něj pak vezmeme pravou stranu, tj. požadovanou numerickou hodnotu.

Seznam vypočtených hodnot citlivosti pro jednotlivé parametry za různých počátečních podmínek nabízí tabulka 4.1. Ve druhém sloupci tabulky najdeme hodnoty $s_{x_i,1\%}$, ve třetím $s_{x_i,10\%}$. Vidíme, že nejcitlivější je počáteční podmínka pro $N0_2$. Počáteční velikost populace $N0_2$ je specifická a velmi citlivá. Pro $N(0) = N0_2$ totiž velikost populace zůstává konstantní, ale pro sebemenší změnu $N(0)$ populace buď vymře, nebo se její velikost ustálí na jiné (vyšší) hodnotě.

Z hodnot lokálních citlivostí zjistíme, které parametry jsou nejcitlivější, a jejichž hodnoty tak potřebujeme znát co nejpřesněji. Současně však musíme mít na paměti závislost lokální citlivosti na daném nastavení (všech) parametrů, kterou můžeme pozorovat i v tabulce 4.1.

Nastavení parametrů:

$S := 200: \quad r := 1: \quad K := 1000: \quad Nkrit := 150: \quad N0_1 := 100: \quad \Delta 1\_r := 0.01: \quad \Delta 10\_r := 0.1$

Citlivost parametru r :

$$res1 := dsolve\left(\left[\frac{d}{dt}N(t) = (r+\Delta 1\_r)\cdot N(t)\cdot\left(1-\frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit}\cdot N(t) & N(t) \le Nkrit \end{cases}, N(0) = N0_1\right], numeric\right):$$

$$res2 := dsolve\left(\left[\frac{d}{dt}N(t) = (r-\Delta 1\_r)\cdot N(t)\cdot\left(1-\frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit}\cdot N(t) & N(t) \le Nkrit \end{cases}, N(0) = N0_1\right], numeric\right):$$

$$res := dsolve\left(\left[\frac{d}{dt}N(t) = r\cdot N(t)\cdot\left(1-\frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit}\cdot N(t) & N(t) \le Nkrit \end{cases}, N(0) = N0_1\right], numeric\right):$$

$reseni1 := rhs(res1(15)[2]): \quad reseni2 := rhs(res2(15)[2]): \quad reseni := rhs(res(15)[2]):$

$$s_{r,\,1\%} := \frac{|reseni1 - reseni| + |reseni2 - reseni|}{2\cdot reseni}\cdot 100$$

$$14.14850116$$

$$res1 := dsolve\left(\left[\frac{d}{dt}N(t) = (r+\Delta 10\_r)\cdot N(t)\cdot\left(1-\frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit}\cdot N(t) & N(t) \le Nkrit \end{cases}, N(0) = N0_1\right], numeric\right):$$

$$res2 := dsolve\left(\left[\frac{d}{dt}N(t) = (r-\Delta 10\_r)\cdot N(t)\cdot\left(1-\frac{N(t)}{K}\right) - \begin{cases} S & N(t) > Nkrit \\ \frac{S}{Nkrit}\cdot N(t) & N(t) \le Nkrit \end{cases}, N(0) = N0_1\right], numeric\right):$$

$reseni1 := rhs(res1(15)[2]): \quad reseni2 := rhs(res2(15)[2]):$

$$s_{r,\,10\%} := \frac{|reseni1 - reseni| + |reseni2 - reseni|}{2\cdot reseni}\cdot 10$$

$$18.76072321$$

Obrázek 4.9: Výpočet citlivosti parametru r, vnitřního koeficientu růstu, v modelu růstu populace pod tlakem nespecializovaného predátora pro $N(0) = N0_1$.

Tabulka 4.1: Lokální citlivosti parametrů modelu růstu populace pod tlakem nespecializovaného predátora.

Parametr, počáteční podmínka	$\Delta = 1\%$	$\Delta = 10\%$
Vnitřní koeficient růstu (r), $N(0) = N0_1$	14.15	18.76
Vnitřní koeficient růstu (r), $N(0) = N0_2$	1.62	1.71
Vnitřní koeficient růstu (r), $N(0) = N0_3$	0.62	0.65
Hladina nasycení (S), $N(0) = N0_1$	19.22	31.79
Hladina nasycení (S), $N(0) = N0_2$	1.62	1.65
Hladina nasycení (S), $N(0) = N0_3$	0.62	0.63
Úživnost prostředí (K), $N(0) = N0_1$	0.23	0.23
Úživnost prostředí (K), $N(0) = N0_2$	0.23	0.23
Úživnost prostředí (K), $N(0) = N0_3$	1.62	1.64
Kritická velikost (N_{krit}), $N(0) = N0_1$	19.19	26.96
Kritická velikost (N_{krit}), $N(0) = N0_2$	0.00	0.00
Kritická velikost (N_{krit}), $N(0) = N0_3$	0.00	0.00
Počáteční velikost ($N(0)$), $N(0) = N0_1$	0.77	0.77
Počáteční velikost ($N(0)$), $N(0) = N0_2$	115.75	12.91
Počáteční velikost ($N(0)$), $N(0) = N0_3$	0.00	0.00

5 Maple

5.1 Úvod

Systém Maple je výkonný program vhodný k řešení komplexních matematických problémů. Patří do skupiny systémů počítačové algebry, umožňuje symbolické i numerické výpočty a slouží především k vytvoření speciálních dokumentů, prezentací, interaktivních výpočetních modulů v prostředí Maple. Systém Maple též obsahuje komponenty podporující výuku matematiky.

Výrobcem systému je kanadská společnost Maplesoft Inc., jejíž webové stránky[31] poskytují široké informace o systému a jeho dalších příbuzných programech jako je MapleSim, MapleNet, Maple T.A. a mnoho dalších nástrojů a programů. Webové stránky mimo jiné obsahují tzv. *Aplikační centrum* (Application Center), z nějž si může každý zaregistrovaný uživatel stáhnout ukázkové programy demonstrující použití systému Maple při řešení mnoha různých matematických i technických problémů. Poskytují i tzv. *Studentské centrum* (Student Center), kde si zaregistrovaný student může stáhnout mnoho studijních materiálů. Dalšími významnými zdroji informací o Maple jsou diskuzní fórum uživatelů Maple[32] a web distributora Maple pro Českou a Slovenskou republiku[33], kde je většina dokumentů v českém jazyce.

V následujícím textu se budeme zabývat dosud poslední verzí Maple 14. Se systémem je možné pracovat několika různými způsoby, které volíme při spuštění programu ze startovacího menu počítače nebo kliknutím na příslušnou ikonu na ploše.

5.1.1 Standardní zápisník (Standard Worksheet)

Grafické uživatelské rozhraní Maple zvané *Standard Worksheet* se spustí ze startovacího menu počítače výběrem položky **Programy** > **Maple 14** > **Maple 14** nebo kliknutím na ikonu **Maple 14** na ploše. Toto prostředí poskytuje veškeré možnosti systému Maple a pomáhá vytvářet elektronické dokumenty (zápisníky) zobrazující matematické výpočty, matematické texty a komentáře spolu s propracovanou počítačovou grafikou. Některé výpočty je možné v zápisníku „schovat“ a nechat „odkryté“ jen nejdůležitější pasáže tak, aby dokument poskytoval uživateli potřebné informace. Jelikož vytvořené dokumenty jsou interaktivní, tj. v jistém smyslu „živé“, může si uživatel sám upravovat předdefinované hodnoty parametrů, vyhodnocovat příkazy, a získávat tak nové výsledky.

Menu zápisníku Maple má tři vodorovné lišty: *hlavní menu* (zcela nahoře), *nástrojovou lištu* (pod hlavním menu) a *kontextovou lištu* (pod nástrojovou lištou), dále *palety* (svislý blok na levé straně[34]) a vlastní pracovní pole – *dokument*, do nějž zadáváme příkazy, texty,

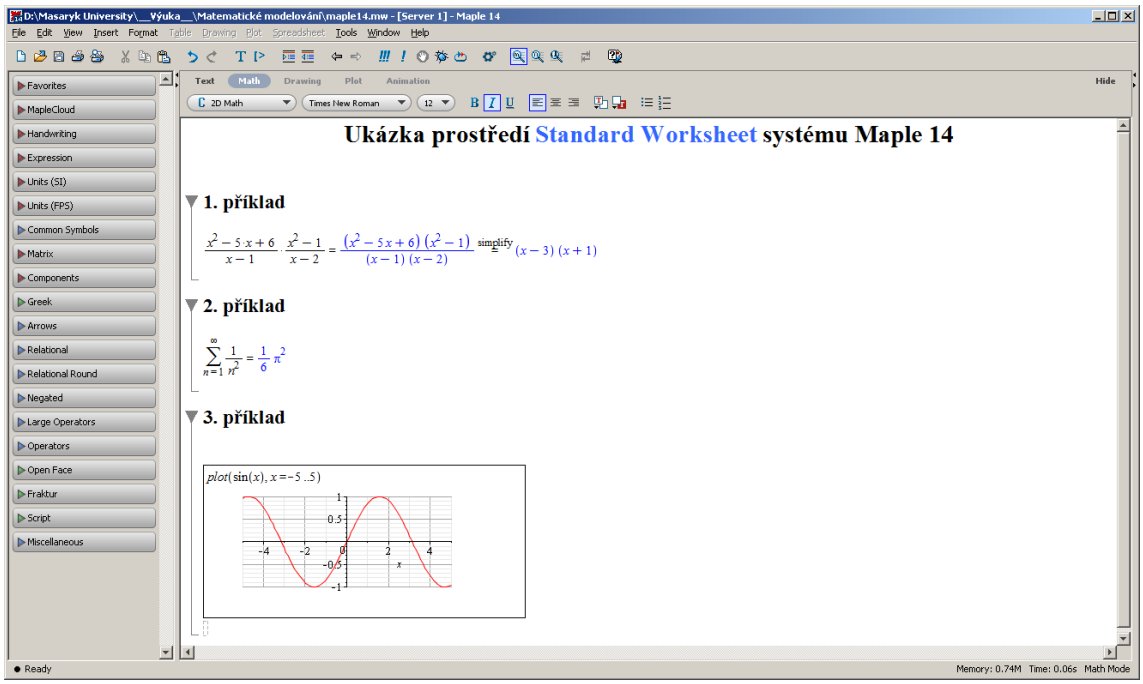
[31] http://www.maplesoft.com

[32] http://www.mapleprimes.com

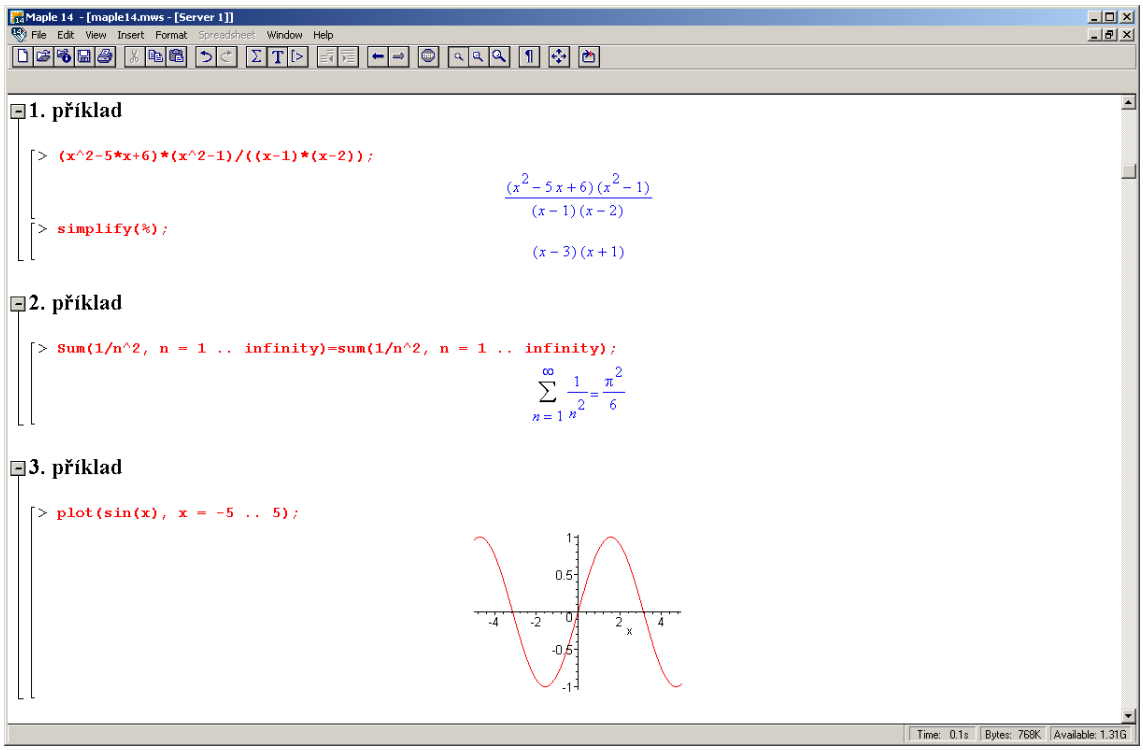
[33] http://www.maplesoft.cz

[34] Palety se automaticky zobrazují v levém bloku. Prostředí systému nabízí blok pro palety i po pravé straně obrazovky (automaticky zavřený), kam je možné některé (ale i všechny) přesunout.

provádímme výpočtové a grafické akce. Vlastní pracovní pole je možné zobrazit přes celou obrazovku skrytím palet, nástrojové lišty a kontextové lišty (kliknutím na příslušné položky v záložce **View** hlavního menu).



Obrázek 5.1: Maple 14: Prostředí Standard Worksheet



Obrázek 5.2: Maple 14: Prostředí Classic Worksheet

5.1.2 Klasický zápisník (Classic Worksheet)

Classic Worksheet se spustí ze startovacího menu počítače výběrem položky **Programy > Maple 14 > Classic Worksheet Maple 14** nebo kliknutím na ikonu **Classic Worksheet** na ploše. Tento zápisník Maple je určen především pro méně výkonné počítače s omezenou pamětí. Neposkytuje také všechny funkce, příkazy a možnosti systému Maple jako *Standard Worksheet*. Před verzí Maple 10 byl tento typ zápisníku jediný možný.

5.1.3 Příkazový řádek a kalkulačka Maple

Se systémem Maple můžeme pracovat i pouze v režimu tzv. *příkazového řádku*, který se spouští ze startovacího menu počítače výběrem položky **Programy > Maple 14 > Command-line Maple 14** a je určen k řešení rozsáhlých a složitých úloh. K dispozici nejsou žádné grafické prvky.

Dále je možné používat (a vytvářet) tzv. *maplety*, tj. grafická uživatelská rozhraní obsahující okénka, textová pole a další vizuální prvky umožňující pouhým klikáním spouštět výpočty. *Kalkulačka* systému Maple je speciální typ mapletu, který je k dispozici pouze pro operační systémy Windows. Spouští se ze startovacího menu počítače, kde se vybere **Programy > Maple 14 > Maple Calculator**.

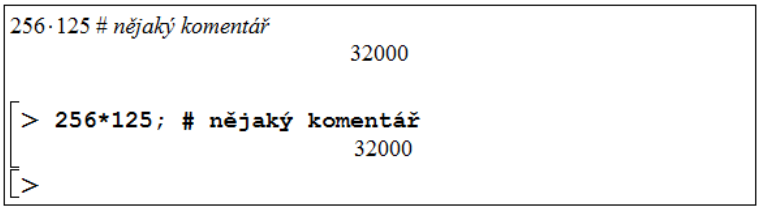
5.1.4 Document Mode, Worksheet Mode

Ve zbývající části kapitoly budeme uvažovat rozšířené prostředí *Standard Worksheet*. V tomto prostředí je možné pracovat ve dvou základních režimech: *Worksheet Mode* a *Document Mode*. Prvně jmenovaný odpovídá prostředí *Classic Worksheet*, v němž je každý příkaz Maple uvozen symbolem [> a musí být ukončen dvojtečkou (výsledek se nezobrazí) nebo středníkem (výsledek se zobrazí na dalším řádku uprostřed). Otevírá se v základním menu jako **New**, kde se vybere **Worksheet mode**. Druhý jmenovaný režim poskytuje přehlednější zápis příkazů a matematických vzorců bez „přebytečných“ symbolů. Při otevření nového souboru v základním menu **New** je automaticky spuštěn právě tento režim *Document mode*.

Zápis v režimu *Document Mode*	Zápis v režimu *Worksheet Mode*
Tento text je v režimu **Text Mode**. Stiskem klávesy **Enter** jsme se dostali na další řádek. Na následujícím řádku se přepneme do režimu **Math Mode**.	Tento text je též v režimu **Text Mode**.
příkaz	*Tento text je v režimu **Math Mode**.*
příkaz	> *příkaz* *příkaz*
nějaký text	> **příkaz;** *příkaz*
nějaký text	> 256·125 32000
256·125	> **256*125;** 32000
32000	
$solve(x^2 - 7x + 12 = 0)$	
4, 3	

Obrázek 5.3: Režimy zápisu.

K příkazům máme možnost zapisovat komentáře, a to uvedením symbolu mřížky (#) před text, který má být komentářem (viz obrázek 5.4).



Obrázek 5.4: Zápis komentáře v příkazovém režimu

Obvykle je zápisník nastaven do jednoho režimu, který je možné zvolit při otevírání nového souboru v hlavním menu (**File** > **New** > **Worksheet mode** nebo **File** > **New** > **Document Mode**). Existuje však i možnost přepínat mezi režimy v rámci jednoho zápisníku, kdy je část vytvořena v jednom režimu, část v jiném. Z *Document Mode* se přepneme do *Worksheet Mode* kliknutím na ikonku [> v nástrojové liště (pod hlavním menu). Naopak z *Worksheet Mode* se do režimu *Document Mode* přepneme výběrem položky v hlavním menu (**Format** > **Create Document block**).

5.1.5 Math Mode, Text Mode

Pro rozlišení příkazů a obyčejného textu v režimu *Document Mode* slouží kontextová lišta zápisníku, kde máme na výběr *Text Mode* a *Math Mode*. *Math Mode* odpovídá příkazům (po stisku klávesy **Enter** dojde k jeho vyhodnocení), v *Text Mode* píšeme texty dokumentu podobně jako např. v textovém editoru Word (po stisku klávesy **Enter** přejdeme na nový řádek bez jakéhokoli vyhodnocení). Volit režim zápisu můžeme buď kliknutím myši (v kontextové liště nad dokumentem jsou uvedeny názvy představující jednotlivé možnosti) nebo výběrem položky v hlavním menu (**Edit** > **Switch to Text/Math Mode**). Totéž lze rychleji provést jedinou klávesou **F5**.

V režimu *Worksheet Mode* lze pro text i pro příkazy použít oba druhy zápisu. Pro psaní textu je nutné kliknout na ikonku **T** v nástrojové liště nebo zvolit položku v hlavním menu (**Insert** > **Text**). Podobně pro zápis příkazů jazyka Maple je nutné kliknout na ikonku [> v nástrojové liště nebo zvolit položku v hlavním menu (**Insert** > **Maple Input**). Při otevření nového souboru je zápisník automaticky nastaven na psaní příkazů.

5.2 Základní ovládání systému

V této části si ukážeme, jak systému Maple zadávat jednoduché příkazy. Budeme proto předpokládat, že zápisník je již nastaven pro psaní příkazů (*Math Mode*).

Základní operace: pro sčítání používáme symbol plus (+), pro odčítání mínus (−), pro násobení (*), ale pozor, pro dělení musíme používat pouze lomítko (/), dvojtečka (:) má jiný význam (viz dále).

Zadání zlomku: zadáme čitatel, lomítko (/) a jmenovatel. Pro opuštění zápisu jmenovatele stačí stisknout šipku doprava (ve zlomku je též možno pohybovat se šipkami).

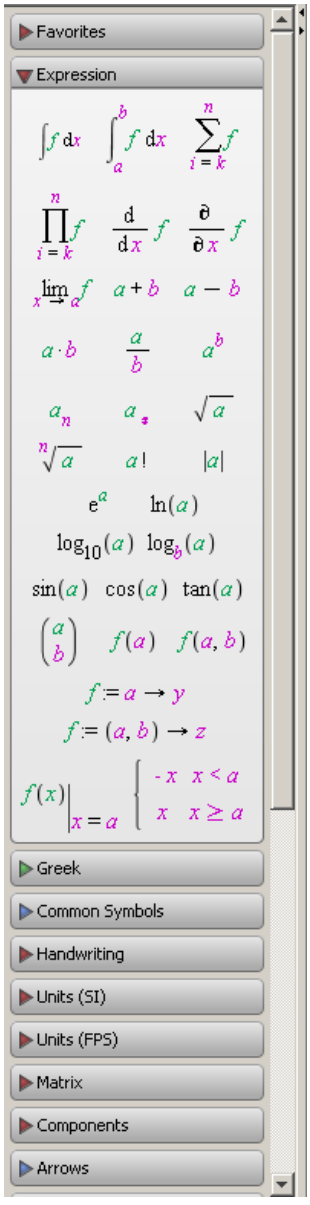
Zadání mocniny: zadáme základ, symbol stříška (^) a exponent. Pro opuštění zápisu exponentu je opět možné použít šipku doprava.

5.2.1 Vyhodnocení příkazů

Příkaz vyhodnotíme stiskem klávesy **Enter**. Výsledek se zobrazí na dalším řádku uprostřed. V dřívějších verzích (méně než 10) systému Maple bylo nutné příkaz ukončovat středníkem, aby se provedl. Tato možnost nadále zůstala (tj. zadáme-li za příkaz středník, „nic nepokazíme“) a v některých situacích je dokonce jediná možná – např. textový režim (*Text Mode*) příkazů v *Worksheet Mode* nebo při psaní příkazů v prostředí *Classic Worksheet*. Z předešlých verzí Maple se uchovala i funkcionalita symbolu dvojtečka (:), která po zařazení za příkaz a následného stisku klávesy **Enter** potlačí zobrazení výsledku na dalším řádku (tj. příkaz se provede, ale na obrazovku se nic nevypíše). Proto není možné dvojtečku používat jako operátor dělení. V *Document Mode* je navíc možné zapisovat příkaz i s výsledkem na jeden řádek. Po napsání příkazu k tomu stačí namísto stisku klávesy **Enter** použít klávesovou zkratku „**Ctrl** + =“.

Jak bylo zmíněno dříve, interaktivní dokumenty v Maple jsou „živé“. Tím máme na mysli skutečnost, že i v již dříve vytvořeném programu s vyhodnocenými příkazy otevřeném po libovolně dlouhé době můžeme kterýkoli výraz upravit, znovu vyhodnotit (stisknout **Enter** nebo „**Ctrl** + =“) a dostaneme nový (správný) výsledek. Označíme-li myší několik (libovolně mnoho) příkazů a stiskneme ikonku **!** (vykřičník) z nástrojové lišty, všechny označené příkazy budou postupně vyhodnoceny. K vyhodnocení všech příkazů v dokumentu slouží ikonka **!!!** (tři vykřičníky).

Maple obsahuje více než tisíc symbolů, pomocí nichž můžeme tvořit matematické výrazy. Patří mezi ně písmena a číslice, jimiž vytváříme jména (posloupnost znaků začínající písmenem, za kterým může následovat kombinace písmen, čísel a vybraných symbolů), reálná čísla (celá, racionální, s desetinou tečkou nebo v notaci pohyblivé řádové čárky), komplexní čísla (sestávající z reálné a imaginární části), aritmetické, booleovské a jiné operátory (+, –, !, /, *, $\int$, lim, ...), konstanty (π, e, ...), imaginární jednotka, nekonečno, matematické funkce ($cos(x)$, $sin(\frac{\pi}{3})$, ...) a proměnné (pojmenované jménem ...). Velkou předností systému Maple je jeho schopnost symbolických matematických výpočtů (tj. práce s výrazy). Některé z matematických symbolů, které můžeme použít, nejsou na klávesnici, a tak se zadávají buď z palety nebo pomocí svých názvů.



Obrázek 5.5: Palety

5.2.2 Palety

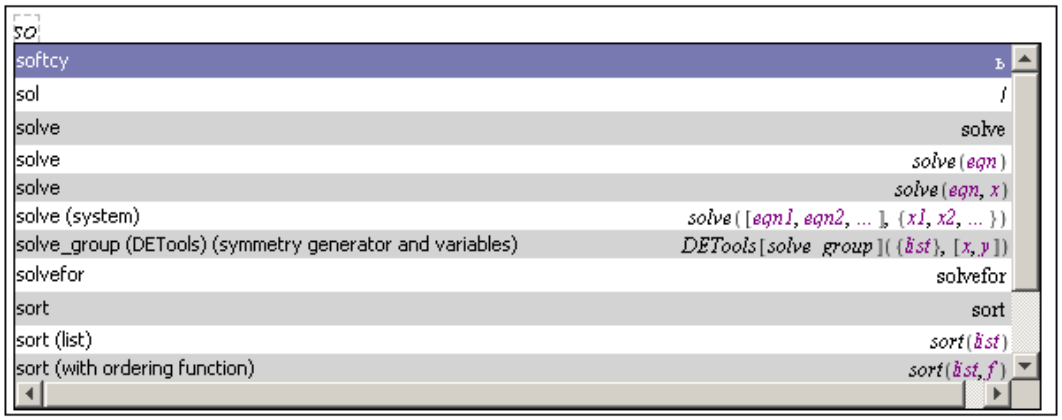
Palety jsou pojmenované „obdélníčky“ s nabídkou předdefinovaných symbolů, zápisů, výrazů apod. (obrázek 5.5), zpravidla při levém okraji zápisníku. Každá paleta obsahuje symboly příslušné skupiny. Například paleta s názvem *Expression* nabízí některé základní matematické výrazy, paleta *Greek* písmena řecké abecedy atd. Standardně zůstává několik palet nezobrazených. V hlavním menu (**View** > **Palettes**) můžeme seznam zobrazených palet upravit tím, že některé přidáme, odebereme, ale třeba i jinak seřadíme. Totéž lze provést jen za pomoci myši. Přidržením levého tlačítka vybranou paletu přesuneme na jiné místo, stisknutím pravého tlačítka vyvoláme stejnou nabídku, jako bychom postupovali přes hlavní menu. Kliknutí levého tlačítka myši na některou z palet zobrazí (příp. skryje) symboly, které paleta nabízí. Vložit z palety symbol

do zápisníku pak stačí pouhým kliknutím, případně „přetáhnutím“ s pomocí levého tlačítka myši. Obecné barevně zvýrazněné symboly ve výrazu je možné dále specifikovat (upravovat) dosazením hodnoty, s níž potřebujeme pracovat.

Nyní ukážeme, jak vložit například výraz $\sum_{i=1}^{10} 2^i$. Pro jeho zapsání potřebujeme celkem dva různé symboly: *sumu* a *mocninu*. Sumační symbol nalezneme v paletě **Expression**. Kliknutím na tuto paletu ji otevřeme (na obrázku 5.5 je jako jediná otevřená) a vybereme z ní přítomný sumační symbol. Po jeho vložení do zápisníku pak jednoduše přepíšeme obecné symboly (k, n a f) požadovanými hodnotami. Pohybovat ve vzorci se můžeme pomocí šipek na klávesnici, s výhodou lze využít klávesy **Tab**, přitom k úpravě výrazu můžeme používat i všechny ostatní klávesy jako např. **Delete**, **Backspace**, **mezerník**, ... Znak f přepíšeme mocninným výrazem. To můžeme provést buď vložením dalšího symbolu z palety **Expression** (symbol a^b) a následnou úpravou (specifikací hodnot a, b), nebo užitím již známé klávesy pro tvorbu mocnin – stříšky (`^`).

5.2.3 Názvy symbolů

Mimo palet můžeme k zápisu symbolů užívat jejich názvů. Například symbol π vložíme zapsáním jeho názvu `Pi`[35], pro odmocninu je vyhrazen název **sqrt**, takže $\sqrt{x}$ vložíme napsáním `sqrt(x)`. Při vkládání symbolů pomocí názvů nebo při tvorbě příkazů může přijít vhod funkce „dokončování“. Ta se vyvolá stisknutím klávesy **Esc**, nebo kláves „**Ctrl** + **mezerník**“. Pro zadání symbolu pak stačí napsat jeho úvodní písmeno (písmena) a pomocí klávesové zkratky následně z vyskakovacího okénka zvolit požadovanou hodnotu. Na obrázku 5.6 je ukázka nabídky pro dokončení zápisu písmen **so**.



Obrázek 5.6: Funkce automatického dokončování.

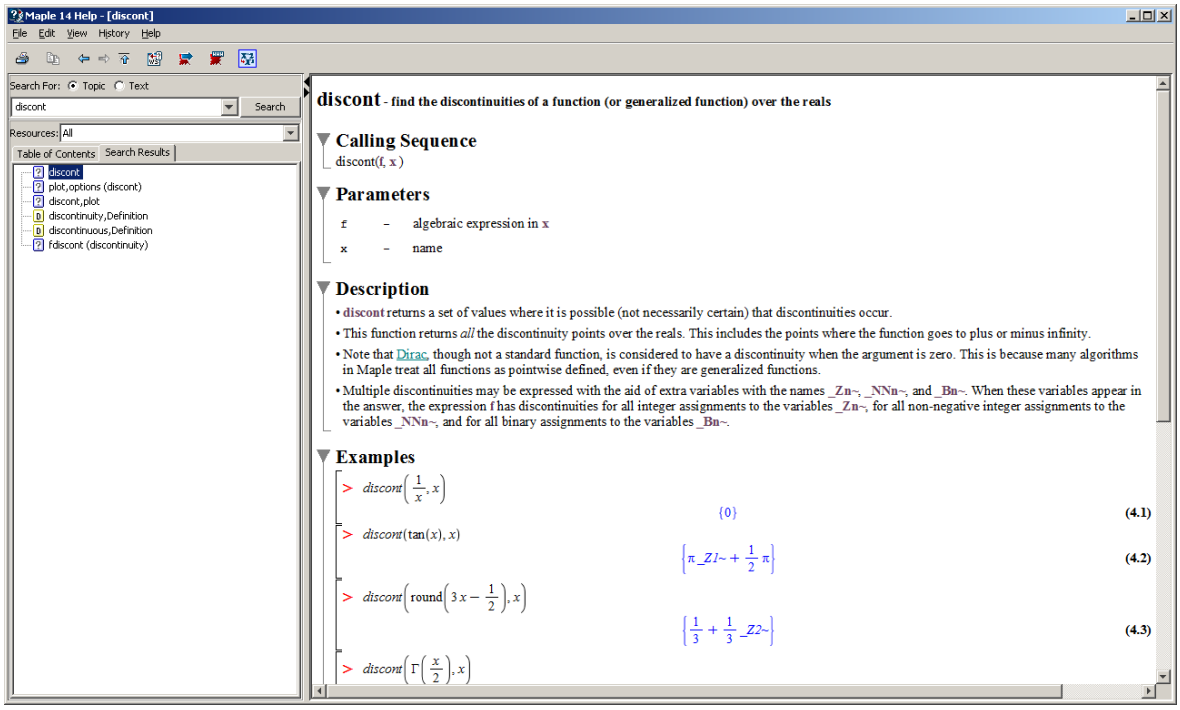
5.3 Nápověda

Nápověda je významnou součástí systému Maple a pomáhá jeho uživateli velmi rychle pochopit a využívat prostředí Maple. K dispozici je několik různých typů nápovědy.

[35] Maple rozlišuje malá a velká písmena. Například zápis `Pi` představuje Ludolfovo číslo π i s jeho hodnotou, zatímco zápis `pi` představuje pouze symbol (řecké písmeno) π.

5.3.1 Maple Help

Vyvoláním **Maple Help** se zobrazí základní stránky nápovědy se všemi dostupnými informacemi. Je v nich možné jednak vyhledávat požadované téma, ale také procházet jednotlivá témata jako v manuálu. Na obrázku 5.7 je okno nápovědy otevřené na informacích k příkazu **discont**.



Obrázek 5.7: Hlavní nápověda systému Maple.

5.3.2 Tour of Maple, Quick Reference, Quick Help

Položka **Take a Tour of Maple** poskytuje interaktivní přehled systému (jeho nejdůležitějších prvků). Kliknutím na **Quick Reference** zobrazíme tabulku informací o ovládání systému Maple, zejména pro nové uživatele. Jedná se o základní informace s odkazy do nápovědy **Maple Help** pro jejich případné doplnění. Položka **Quick Help** nabízí ještě stručnější tabulku než předchozí nápověda. Standardně se objevuje v každém novém zápisníku při pravé straně v podobě černého okénka (pokud toto nastavení nezrušíme). Po zavření je možné ji vyvolat stiskem klávesy **F1**, či jako položku v hlavním menu.

5.3.3 What's New, Startup Dialog

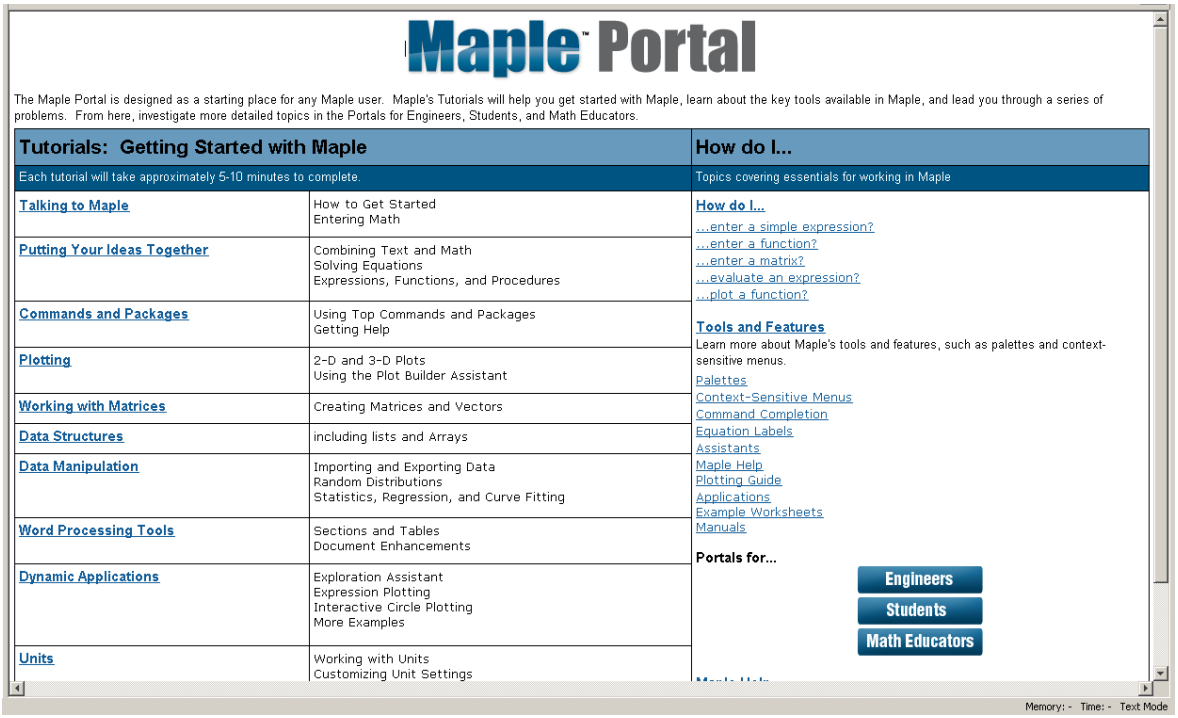
K dispozici je též přehled rozšíření stávající verze Maple oproti předcházející verzi (dostupné přes **Help** > **What's New**). Nápověda **Startup Dialog** obsahuje tipy pro práci se systémem Maple. Zobrazuje se vždy po spuštění systému (pokud toto nastavení nezrušíme).

5.3.4 Manuals, Resources, and more

Vyvoláním **Manuals, Resources, and more** přejdeme do další oblasti nápovědy, z níž popíšeme tři nejdůležitější části.

Maple Portal

Od předcházející verze (tedy Maple 13) je k dispozici tzv. *Maple Portal*. Spustit jej můžeme samostatně (*Maple Portal* má vlastní ikonu na ploše) nebo přes nápovědu v hlavním menu (**Help** > **Manuals, Resources, and more** > **Maple Portal**). *Maple Portal* slouží jako pomocník všem novým (a nejen těm) uživatelům systému Maple. Je v něm možné rychle najít detailní popis práce se systémem Maple od řešení nejjednodušších problémů až po velmi složité úlohy.



Obrázek 5.8: Maple Portal.

Applications and Examples

Z nápovědy je možno vyvolat i spustitelné soubory (tj. již vytvořené programy) demonstrující možnosti systému Maple. Vyvoláme je přes nápovědu v hlavním menu (**Help** > **Manuals, Resources, and more** > **Applications and Examples**) a pak kliknutim na zvolený příklad.

Manuals

Z nápovědy je možno vyvolat i anglické manuály **User manual**, **Introductory Programming Guide**, **Advanced Programming Guide** a **Getting Started with Maple Toolboxes** podrobně popisující možnosti systému Maple. Vyvoláme je přes nápovědu v hlavním menu (**Help** > **Manuals, Resources, and more** > **Manuals**) a pak kliknutim na zvolený manuál.

5.3.5 Pomocníci, instruktoři a řešené úlohy

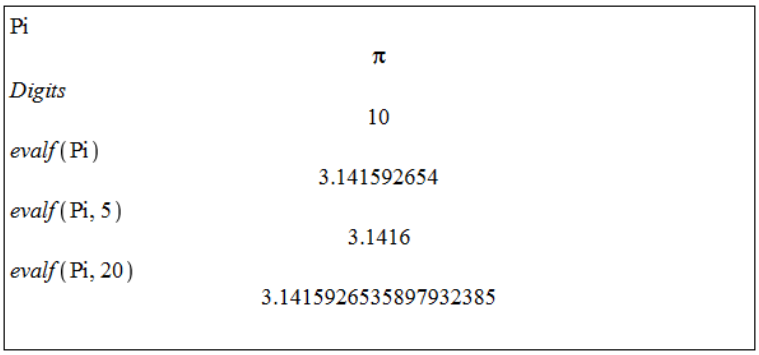
Systém Maple poskytuje také již připravené „pomocné nástroje“ pro řešení úloh. Jsou to tzv. *Pomocníci* (*Assistants*), *Instruktoři* (*Tutors*) a *Úlohy* (*Tasks*), které vyvoláme z hlavního menu (**Tools** > **Assistants** nebo **Tools** > **Tutors** anebo **Tools** > **Tasks**). *Pomocníci*

(*Assistants*) mají např. nástroje pro hledání funkční závislosti v datech, optimalizaci funkcí, řešení diferenciálních rovnic a další. Pro daný typ úlohy mají implementováno několik často používaných algoritmů. Po vyvolání provedou uživatele nastavením a specifikací parametrů úlohy a zvolenou metodou úlohu vyřeší. *Instruktoři* (*Tutors*) provedou uživatele řešenou problematikou pomocí jednoduchých názorných příkladů. *Úlohy* (*Tasks*) zobrazují na příkladech, jak řešit různé úlohy. Zobrazí se vyvoláním z menu (**Tools** > **Tasks** > **Browse**).

5.4 Provádění výpočtů

Maple provádí přesně numerické výpočty s celými, racionálními i iracionálními čísly. Každý zadaný matematický výraz se snaží co nejvíce zjednodušit (např. zlomek zkrátit a převést na základní tvar, upravit algebraický výraz, ...), ale ne za cenu ztráty přesnosti. To znamená, že například racionální čísla (zlomky) udržuje stále v jejich základním tvaru. Podobně s konstantami π, e a dalšími, s odmocninami a jinými výrazy pracuje jako se symboly. Tímto je zaručena absolutní přesnost numerických výpočtů i v případě, kdy nepracujeme pouze s celými čísly.

Jsou však situace, kdy potřebujeme znát přibližnou hodnotu reálného nebo racionálního čísla v pohyblivé řádové čárce. K tomu slouží příkaz **evalf**, jenž vrátí zaokrouhlenou hodnotu svého argumentu na počet platných cifer mantisy specifikovaný systémovou proměnnou **Digits**. Ta je standardně nastavena na hodnotu 10. Všechny výpočty, při nichž je nutné zaokrouhlovat čísla, provádí proto Maple s přesností na 10 platných míst. Proměnnou **Digits** můžeme nastavit na takřka libovolné přirozené číslo. Omezení, jak vysoké toto číslo může být, zjistíme příkazem **kernelopts(maxdigits)**. Pro představu uveďme, že pro Maple 14 je toto číslo 268 435 448, tedy více než 268 milionů platných cifer, s kterými dokáže systém počítat.



*Obrázek 5.9: Příkaz **evalf**.*

Aniž bychom měnili nastavení proměnné Digits, můžeme zobrazit libovolný výraz s požadovanou přesností pouze pomocí příkazu **evalf**. Příkaz je možné použít s jedním nebo dvěma parametry. Jediný zadaný parametr znamená, že tento zadaný výraz bude vyhodnocen na počet platných míst specifikovaný v proměnné **Digits**. Přidaný druhý parametr řekne funkci **evalf**, na kolik platných míst má výraz vyhodnotit, viz příklady na obrázku 5.9.

S dříve zmíněnou nápovědou souvisí symbol **?** (otazník). Otazník spolu s názvem příkazu otevře nápovědu na stránce týkající se zadaného příkazu. Tedy např. příkaz `?evalf` otevře hlavní nápovědu systému na stránce popisující syntaxi a sémantiku příkazu **evalf** spolu s příklady jeho použití. Zapsáním dvou otazníků na začátek příkazu otevřeme tutéž stránku nápovědy ve „sbaleném“ tvaru osnovy, v níž je možné otevřít (odkrýt) libovolné

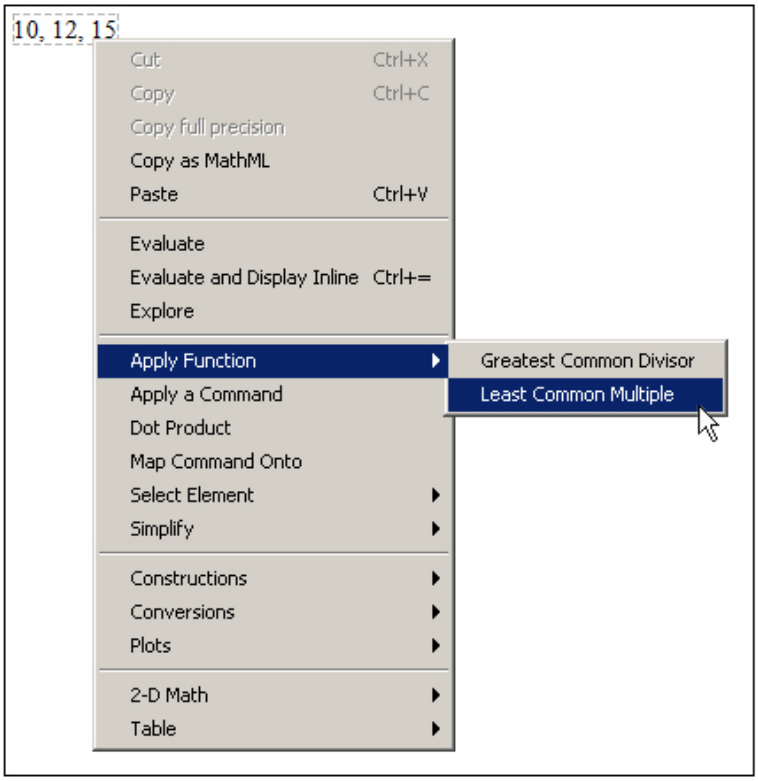
části. Nakonec je tu ještě možnost zadání tří otazníků před příkaz, což otevře nápovědu na příkladech použití tohoto příkazu.

Maple rozeznává přesná čísla (mezi něž patří i zmíněné symboly π a e, zlomky atp.) a čísla typu *Floating-Point*, nebo-li čísla v pohyblivé řádové čárce. Jestliže systému zadáme výraz, v němž některý z jeho podvýrazů bude typu *Floating-Point*, může Maple na celý výraz pohlížet jako by byl tohoto typu a bude výsledky výpočtů zaokrouhlovat. To nejlépe uvidíme na dalších příkladech na obrázku 5.10.

$$\frac{2}{3}+\frac{3}{2}=\frac{13}{6}, \quad \text{ale} \quad \frac{2}{3}+1.5=2.166666667$$

$$\sqrt{2}+\sqrt{3}=\sqrt{2}+\sqrt{3}, \quad \text{ale} \quad \sqrt{2.0}+\sqrt{3}=1.414213562+\sqrt{3}$$

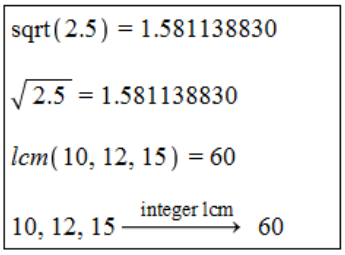
Obrázek 5.10: Přesná čísla a čísla typu ***Floating-Point****.*



Obrázek 5.11: Provedení výpočtu pomocí kontextové nabídky.

5.4.1 Příkazy

Pro provedení výpočtu máme zpravidla více možností. Tou základní, která je k dispozici ve všech verzích systému, jsou příkazy jazyka Maple. Chceme-li například vypočítat odmocninu z čísla 2.5, zapíšeme v systému Maple příkaz `sqrt(2.5)`. Stejného výsledku dosáhneme použitím symbolu pro odmocninu z palety **Expression**. Pokud chceme určit nejmenší společný násobek čísel 10, 12 a 15, můžeme využít příkazu **lcm**, nebo zapsat čísla na řádek za sebe (oddělená čárkami) a přes pravé tlačítko myši zvolit z kontextové nabídky **Apply Function** > **Least Common Multiple**, viz obrázky 5.11, 5.12.

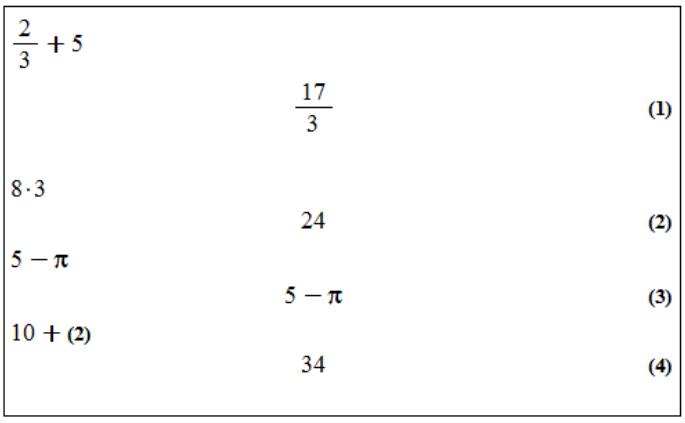


Obrázek 5.12: Různé možnosti v provedení výpočtu.

5.4.2 Označení výsledků

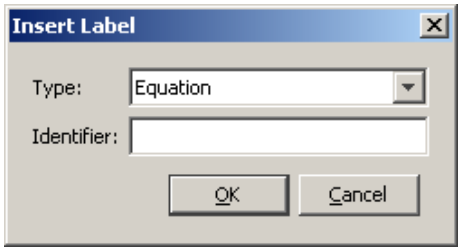
Každému zobrazenému výsledku se v zápisníku přiřazuje číselné označení, které se zapisuje zcela vpravo na řádek s odpovídajícím výsledkem. Označení je možné potlačit (tj. nezobrazovat), znovu vyvolat, případně upravit jeho formát v hlavním menu (**Format** > **Equation Labels** > **...**). Díky označení se můžeme na předešlé výsledky odvolávat a používat je při tvorbě dalších příkazů. V ukázkách vytvořených dokumentů (prezentovaných v tomto textu) je označení výsledků vždy potlačeno. Použití označení ilustruje obrázek 5.13.

Pokud chceme například přičíst číslo **10** k výsledku s označením **(2)**, pak napíšeme „`10 + `“ a přes klávesovou zkratku „**Ctrl** + **L**“ vložíme požadované označení (tedy do „vyskakujícího okénka“ zadáme číslo `2` a potvrdíme (**OK**)). Místo klávesové zkratky „Ctrl + L“ je možné použít horní menu (**Insert** > **Label...**). Upozorněme, že zápis `(2)` vytvořený (pouze) na klávesnici při tvorbě příkazu Maple nepochopí, pro vložení označení do příkazu je třeba důsledně používat předešlý postup s „vyskakujícím okénkem“ zobrazeným na obrázku 5.14.

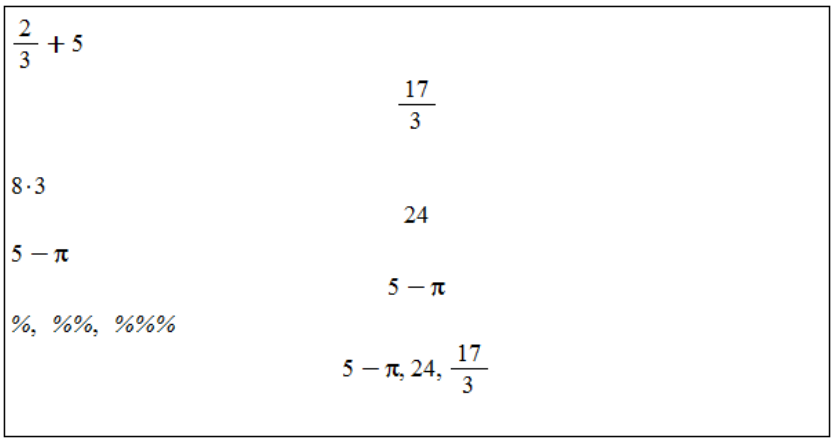


Obrázek 5.13: Označení výsledků.

Maple dále nabízí možnost odkazovat se na poslední tři výsledky (v tomto případě je jedno, zda byly zobrazeny či nikoliv, a zda mají nějaké označení) pomocí symbolu % (procento). Jedno procento (%) představuje poslední výsledek, dvě procenta (%%) předposlední a tři procenta (%%%) před-předposlední. Upozorněme, že výsledek získaný těmito příkazy závisí na pořadí vykonaných příkazů, ne na jejich umístění v zápisníku! Tedy např. % vypíše poslední výsledek získaný předchozím (časově) vykonaným příkazem.



Obrázek 5.14: „Vyskakující okénko“ pro zadání označení.



Obrázek 5.15: Využití procent při odkazování se na předchozí výsledky.

5.4.3 Přiřazení hodnot do proměnných

Odkazovat se na výrazy můžeme také po jejich přiřazení k nějaké proměnné. Operátorem přiřazení je (dvoj)symbol := (dvojtečka + rovnítko).

Namísto (dvoj)symbolu := můžeme k přiřazení použít příkaz **assign**. Tak, jak můžeme výrazy do proměnných přiřazovat, můžeme též přiřazení zrušit (tj. odebrat proměnné uloženou hodnotu). Zmíněné provedeme příkazem **unassign** nebo přiřazením názvu proměnné v apostrofech (obrázek 5.16).

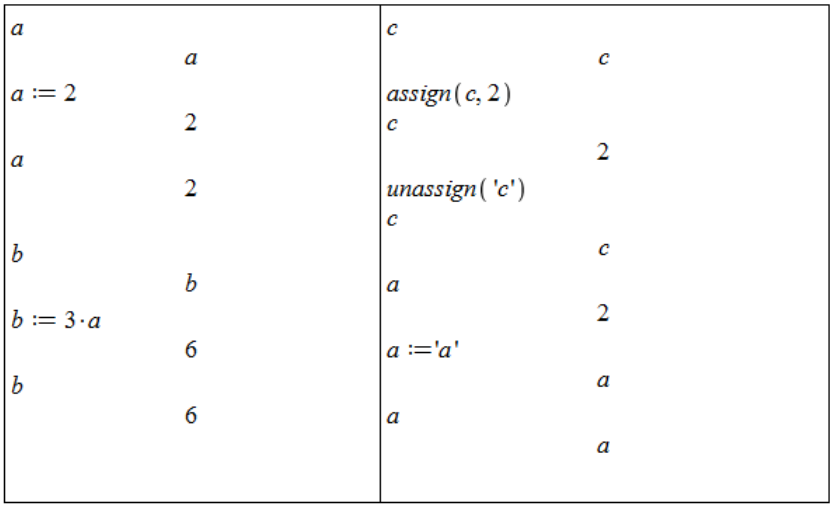
Přiřazovat hodnoty můžeme i do tzv. systémových proměnných. Již jsme se setkali s proměnnou **Digits**, která vyjadřuje počet platných míst, s nimiž Maple počítá. Ilustraci na obrázku 5.17 můžeme srovnat s obrázkem 5.9.

Odstranit uloženou hodnotu v systémové proměnné nelze. Do systémových proměnných můžeme hodnoty pouze přiřazovat, nebo současně vrátit příkazem **restart** nastavení všech systémových proměnných na jejich původní hodnoty. Provedení příkazu odstraní všechny uložené hodnoty v paměti (tedy i námi definované proměnné, načtené balíky atd.). Příkaz **restart** se proto používá zpravidla na počátku řešení nové úlohy, zejména pak na začátku každé práce se zápisníkem (aby se předešlo tomu, že budeme používat proměnnou, v níž je z dřívějška uložena pro nás nesprávná hodnota).

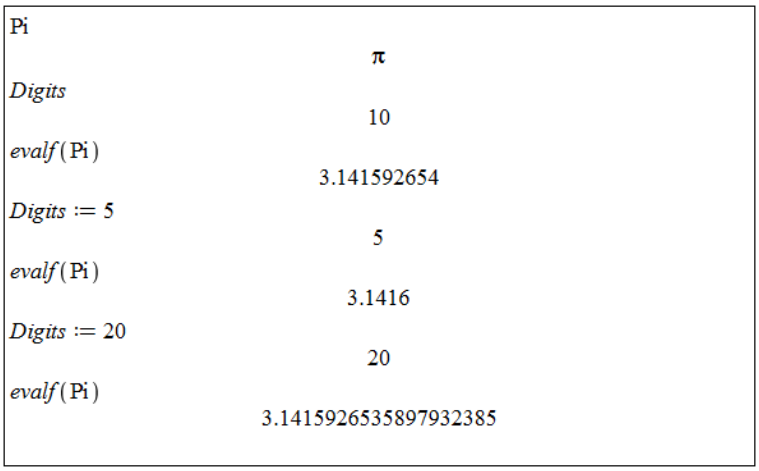
5.4.4 Balíky

Knihovna příkazů jazyka Maple je rozdělena na hlavní knihovnu a tzv. balíky[36]. Příkazy, s nimiž jsme se doposud setkali, patří do hlavní knihovny, a můžeme je tak používat ihned

[36]Kromě pojmu balík se v češtině používá také termín knihovna.



Obrázek 5.16: Přiřazení hodnot do proměnných a odstranění uložené hodnoty.



*Obrázek 5.17: Proměnná **Digits** a příkaz **evalf**.*

po spuštění systému. Naproti tomu většina příkazů naleží do balíků, které musíme před použitím příslušného příkazu buď načíst do dokumentu pomocí příkazu **with**, nebo zadat příkaz spolu s názvem balíku. Načtení balíku pomocí příkazu **with** umožní používání všech příkazů z příslušného balíku. Naopak zadání příkazu spolu s názvem balíku je nutné provádět při každém použití tohoto příkazu, pokud balík nenačteme (příkazem **with**).

Například příkazy pro práci s vektory a maticemi náleží do balíku **LinearAlgebra**. Jestliže chceme tedy použít příkaz **Eigenvalues** pro nalezení vlastních čísel matice, načteme nejprve balík **LinearAlgebra**, jak dokumentuje obrázek 5.18.

Jednotky

Práci s jednotkami umožňuje balík **Units**. Při výpočtech tak nemusíme pracovat jen s čísly, ale můžeme jim přiřazovat i jednotky. K vložení jednotek do zápisníku využijeme palety **Units**. Obrázek 5.19 ilustruje použití jednotek při výpočtu gravitační síly působící v tíhovém poli Země (tj. gravitační zrychlení je rovno přibližně 9.81 ms^{-2}) na těleso o hmotnosti 10 kg. Vidíme, že Maple umí jednotky také zjednodušovat (resp. upravovat na jiný tvar). Ke zjednodušení výrazů přitom slouží příkaz **simplify**.

$$Matice := \begin{bmatrix} 1 & 2 & 1 \\ 6 & -1 & 0 \\ -1 & -2 & -1 \end{bmatrix} :$$

$with(LinearAlgebra) :$

$Eigenvalues(Matice)$

$$\begin{bmatrix} 0 \\ 3 \\ -4 \end{bmatrix}$$

Obrázek 5.18: Použití balíků.

$$F := 10[\![kg]\!] \cdot 9.81 \frac{[\![m]\!]}{[\![s]\!]^2}$$

$$\frac{98.10\,[\![kg]\!]\,[\![m]\!]}{[\![s]\!]^2}$$

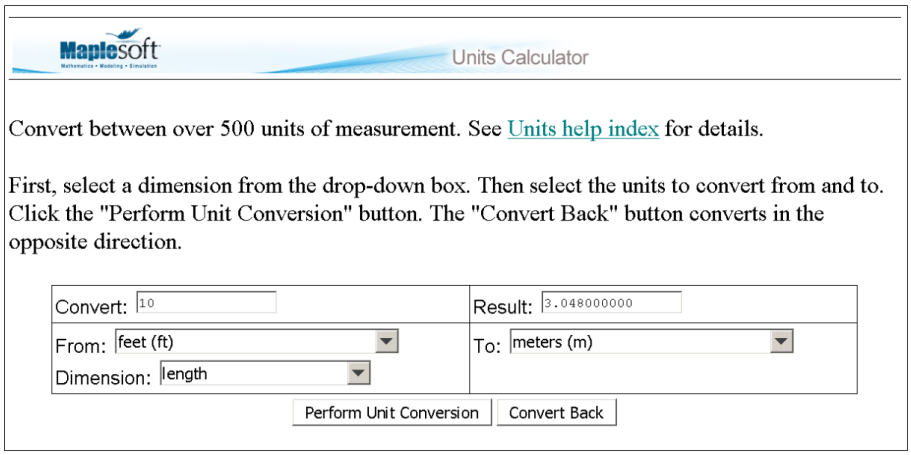
$simplify(\%)$

$$98.10\,[\![N]\!]$$

Obrázek 5.19: Použití jednotek.

Maple rozpoznává jednotky různých soustav a velikostí, s nimiž umí pracovat a vzájemně je převádět. Pro převod jednotek je k dispozici speciální nástroj zvaný **Units Calculator**. Spustit jej můžeme z hlavního menu přes **Tools** > **Assistants** > **Units Calculator...**, ukázku poskytuje obrázek 5.20.

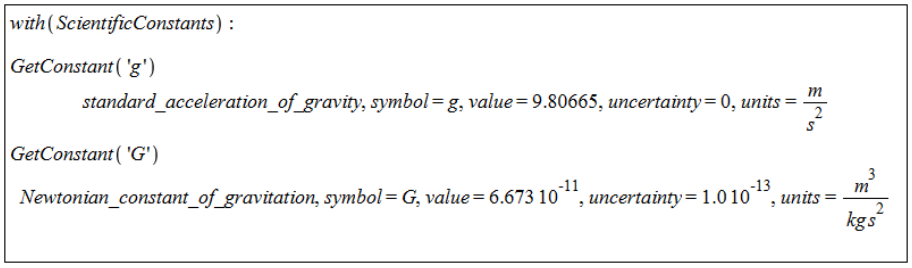
Pokud chceme použít jednotku, která není v paletě **Units**, můžeme si ji vytvořit sami tak, že přidáme jednotku s názvem **unit** a název přepíšeme. V systému Maple 13 je implementováno celkem 568 jednotek (tzn. v paletě **Units** je pouze několik vybraných).



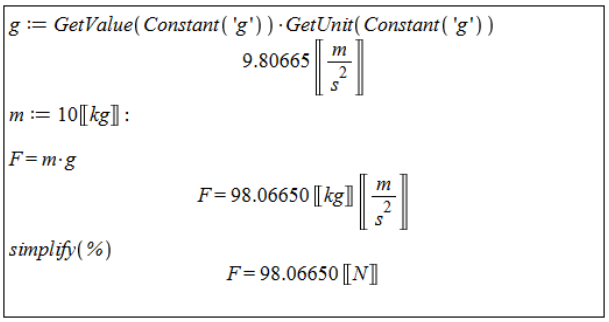
*Obrázek 5.20: **Units Calculator**.*

Vědecké konstanty

Balík **ScientificConstants** poskytuje hodnoty fyzikálních konstant a chemických vlastností látek spolu s jednotkami a neurčitostmi v těchto hodnotách. Příkazem **GetConstant** zobrazíme informace o dané konstantě, viz obrázek 5.21 pro gravitační zrychlení v tíhovém poli Země a Newtonovu gravitační konstantu.



*Obrázek 5.21: Použití balíku **ScientificConstants**.*



Obrázek 5.22: Práce s vědeckými konstantami.

Příkazem **GetValue** získáme (pouze) numerickou hodnotu konstanty, příkazem **GetUnit** (pouze) jednotku – obrázek 5.22.

5.4.5 Řešení rovnic

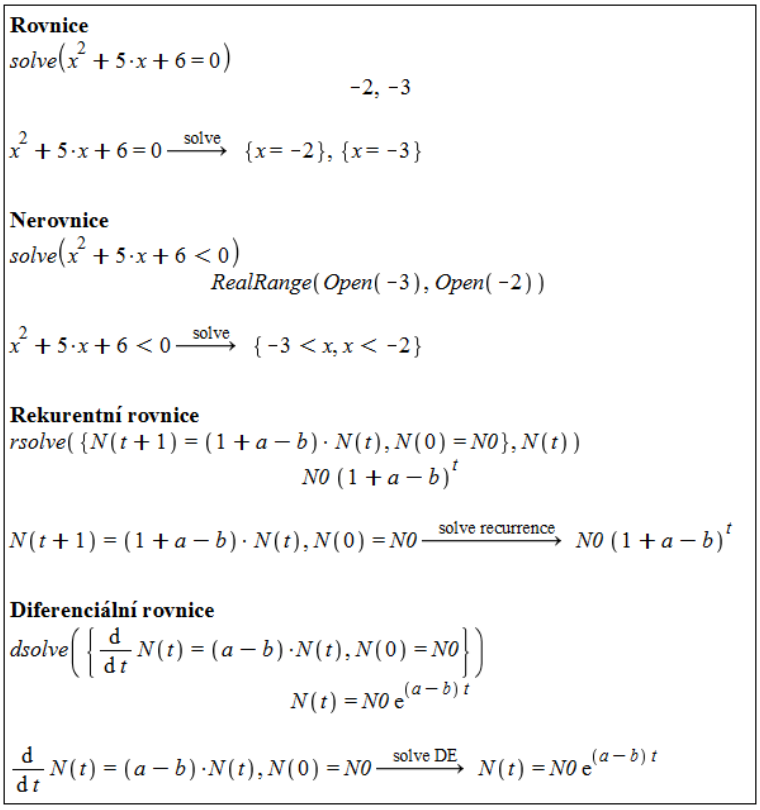
Pro řešení rovnic má systém Maple příkaz **solve** a několik příkazů k němu příbuzných závislých na typech rovnic, viz tabulka 5.1.

Tabulka 5.1: Příkazy pro řešení rovnic

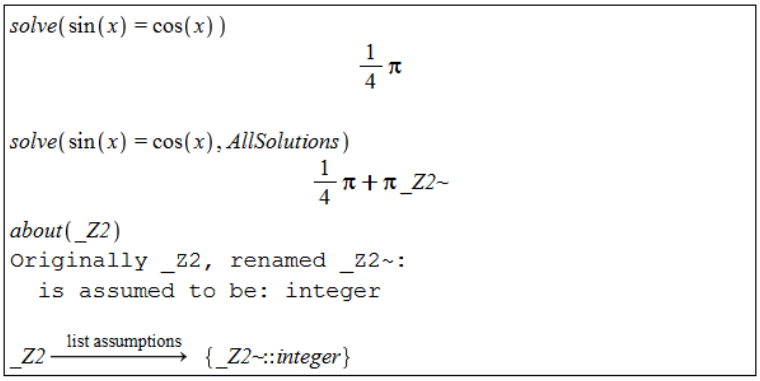
Typ rovnice	**Příkaz pro řešení**
Rovnice a nerovnice	solve, fsolve
Obyčejné diferenciální rovnice	dsolve
Parciální diferenciální rovnice	pdsolve
Rovnice v oboru celých čísel	isolve
Rovnice v oboru celých čísel v konečném tělese	msolve
Lineární integrální rovnice	intsolve
Systémy lineárních rovnic	LinearAlgebra[LinearSolve]
Rekurentní rovnice	rsolve

Pomocí interaktivního prostředí *Standard Worksheet* můžeme řešit rovnice též pomocí kontextové nabídky. Zapíšeme rovnici a pravým tlačítkem myši zvolíme požadovaný příkaz. Obrázek 5.23 ilustruje některé příklady řešení rovnic.

Příkazy pro řešení rovnic nemusí vždy zobrazit všechna řešení. Pokud je chceme zobrazit, přidáme příkazu **solve** nepovinný parametr **AllSolutions**, viz obrázek 5.24.



Obrázek 5.23: Ukázka řešení různých druhů rovnic použitím jednak příkazu, jednak kontextové nabídky.

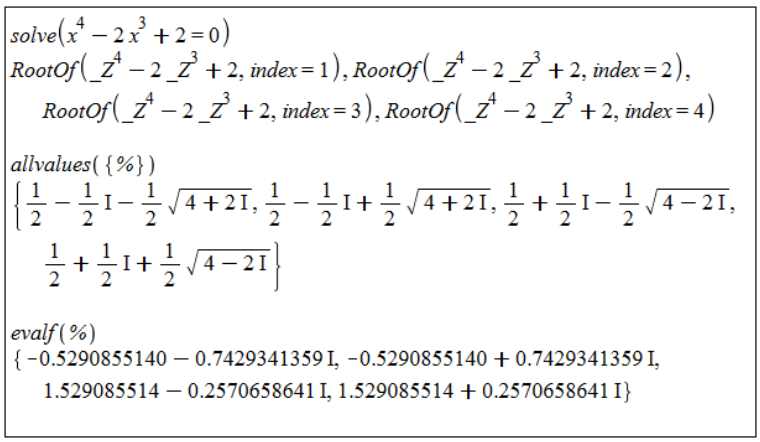


Obrázek 5.24: Zobrazení všech řešení rovnice.

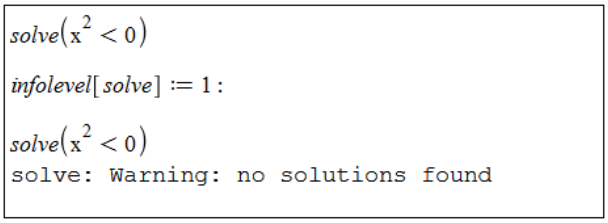
Symbol **_Z2~** na obrázku 5.24 představuje libovolnou celočíselnou proměnnou. Že jde o celočíselnou proměnnou poznáme podle toho, že se v symbolu vyskytuje písmeno **Z**. Podobně by výskyt například písmena **C** značil proměnnou komplexní. Cifra **2** v symbolu

proměnné označuje pořadí, v jakém byla proměnná v zápisníku zavedena. A nakonec znak $\sim$ vyjadřuje, že proměnná splňuje nějaký předpoklad. Jaké předpoklady proměnná splňuje přitom zjistíme příkazem **about**, případně zápisem proměnné a po kliknutí pravým tlačítkem myši zvolením **What Assumptions** z kontextové nabídky. V zobrazeném příkladu na obrázku 5.24 je předpoklad celočíselnosti proměnné přebytečný.

Dále může příkaz **solve** zobrazit výsledek se strukturou **RootOf** vyjadřující kořen (tj. řešení) rovnice v nevyhodnoceném tvaru. Řešení pak vyhodnotíme buď příkazem **allvalues** (pro symbolické vyjádření), nebo příkazem **evalf** (pro numerické vyjádření) – obrázek 5.25. Vedle příkazů můžeme též využít pravého tlačítka myši, zvolit z kontextové nabídky položku **All Values** (pro symbolické vyjádření) a získaný výsledek převést na numerickou hodnotu zvolením **Approximate** > **10** (pro 10 platných míst) z kontextové nabídky.



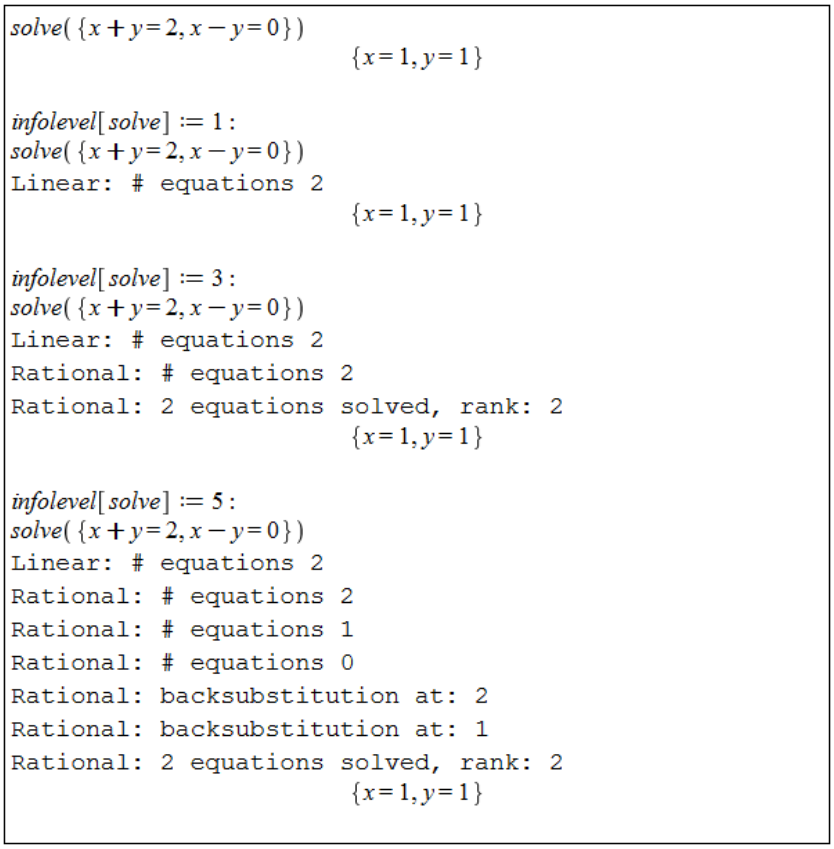
Obrázek 5.25: Tvar zobrazení řešení rovnice.



*Obrázek 5.26: Proměnná **infolevel** a „prázdný výpis“ příkazu **solve**.*

Symboly **_Z** ve struktuře **RootOf** nyní nepředstavují celočíselnou proměnnou (neboť za písmenem **Z** nenásleduje číslo), nýbrž proměnnou libovolnou (tj. i komplexní). Struktura **RootOf** z obrázku 5.25 tedy zastupuje kořen rovnice $z^4 - 2 \cdot z^3 + 2 = 0$.

Systém Maple po zadání příkazu vypíše zpravidla pouze řešení, případně chybová hlášení či varování. U příkazu **solve** (a nejen u něj) toto chování způsobuje „prázdný výpis“ v případě, že Maple žádné řešení nenašel. Pro výpis podrobnějších informací o průběhu vyhodnocení příkazu a výsledcích slouží proměnná **infolevel**. Můžeme ji nastavit buď pro každý příkaz samostatně, přičemž do hranatých závorek za proměnnou vložíme název příslušného příkazu, nebo ji nastavíme všem příkazům současně na stejnou hodnotu uvedením slova **all** do hranatých závorek. Proměnná může nabývat hodnot 1, 2, ..., 5. Čím vyšší hodnota je přiřazena v proměnné **infolevel**, tím více informací o vyhodnocení příkazu obdržíme. Standardně není proměnná nastavena na žádnou hodnotu, což v podstatě odpovídá nastavení proměnné na hodnotu 0. Použití proměnné **infolevel** dokumentují obrázky 5.26 a 5.27.



*Obrázek 5.27: Proměnná **infolevel** a příkaz **solve**.*

5.5 Funkce

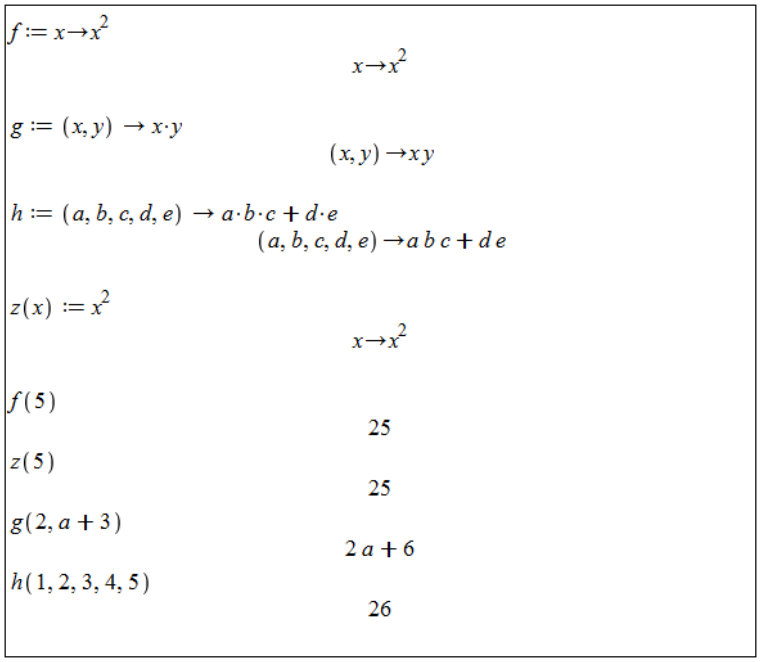
Definovat funkci můžeme různými způsoby. Při zadávání (pouze) na klávesnici napíšeme název funkce, symbol pro přiřazení (**:=**), argument funkce, šipku vpravo a funkční předpis. Šipku přitom vytvoříme jako pomlčku následovanou uzavírající lomenou závorkou (symbol „větší než“). Chceme-li tedy vytvořit funkci $f(x) = x^2$, zapíšeme do zápisníku příkaz `f:=x->x^2`.

Pro zjednodušení můžeme využít palet. Buď je možné při vytváření příkazu použít šipku z palety **Arrows**, nebo můžeme vzít celou šablonu příkazu vytvoření funkce z palety **Expression** a modifikovat v ní požadované symboly. Vytvářet přitom můžeme funkce libovolného počtu proměnných. V prostředí *Standard Worksheet* (s nímž celou dobu pracujeme) můžeme funkci vytvořit též zápisem bez šipky: `f(x):=x^2`. Po odkliknutí příkazu musíme v následně zobrazeném vyskakujícím okénku potvrdit, že se jedná o definici funkce.

Funkční hodnotu v daném bodě získáme zápisem názvu funkce spolu s hodnotami parametrů v závorce (nemusíme přitom zadávat pouze numerické hodnoty).

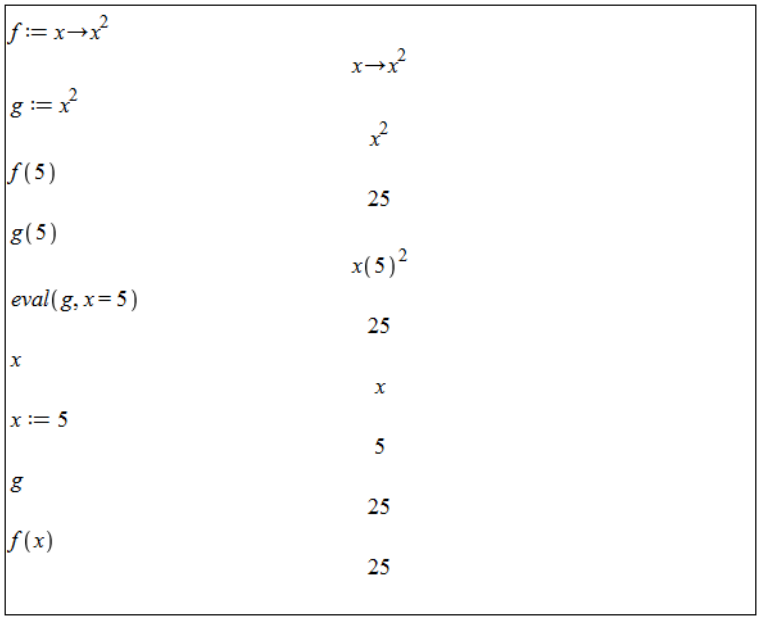
Důležité je rozeznávat funkce od výrazů. Jestliže vytvoříme výraz, například `x^2`, a přiřadíme jej k nějaké proměnné, např. g, jedná se stále pouze o výraz. Hodnotu g pro $x = 5$ nemůžeme proto určit jako funkční hodnotu v bodě 5, ale musíme použít vyhodnocovacího příkazu **eval**, případně do x přiřadit hodnotu 5, viz obrázek 5.29.

Funkce (i výrazy) můžeme též definovat po částech pomocí příkazu **piecewise**. Argumenty v závorce signalizují vždy nejprve interval následovaný funkční hodnotou na tomto intervalu. Poslední interval již zapisovat nemusíme, stačí funkční hodnota, Maple ji doplní ve zbývající množině zatím nedefinovaných bodů. Je možné též sestrojit funkci, která je de-



Obrázek 5.28: Definice funkce.

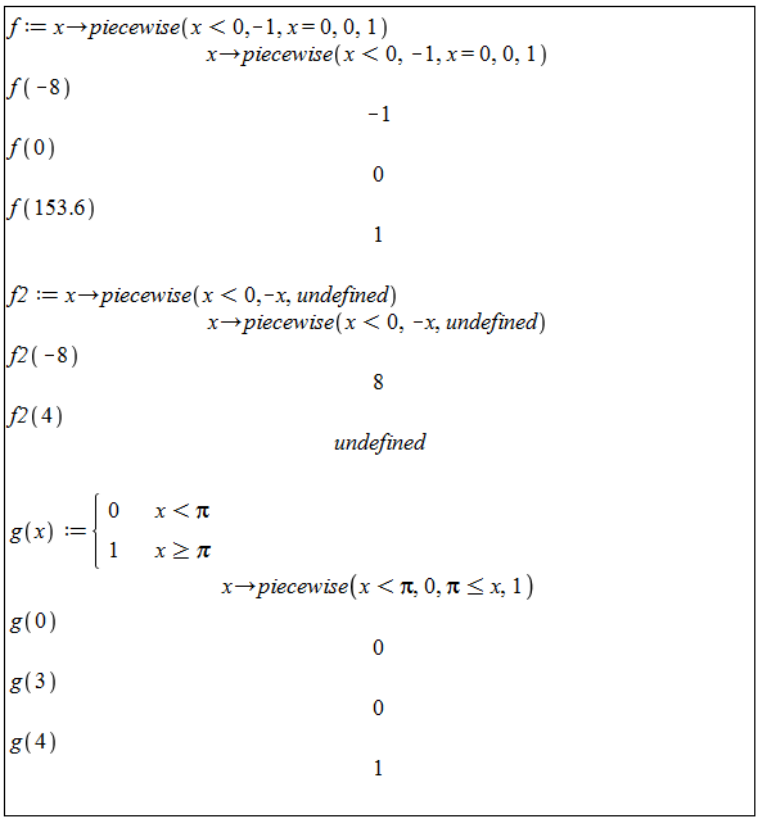
finována pouze na libovolné podmnožině reálných čísel. Pokud má funkce definovaná po částech pouze dva různé předpisy, můžeme k jejímu vytvoření využít symbolu otevřené složené závorky z palety **Expression**.



Obrázek 5.29: Rozdíl mezi funkcí a výrazem.

5.6 Kreslení a animace

Interaktivní prostředí *Standard worksheet* nabízí hned několik možností k vykreslení grafu funkce. Nejrychlejší a zřejmě nejjednodušší je zadat do zápisníku výraz z funkčního předpisu,



Obrázek 5.30: Funkce definovaná po částech.

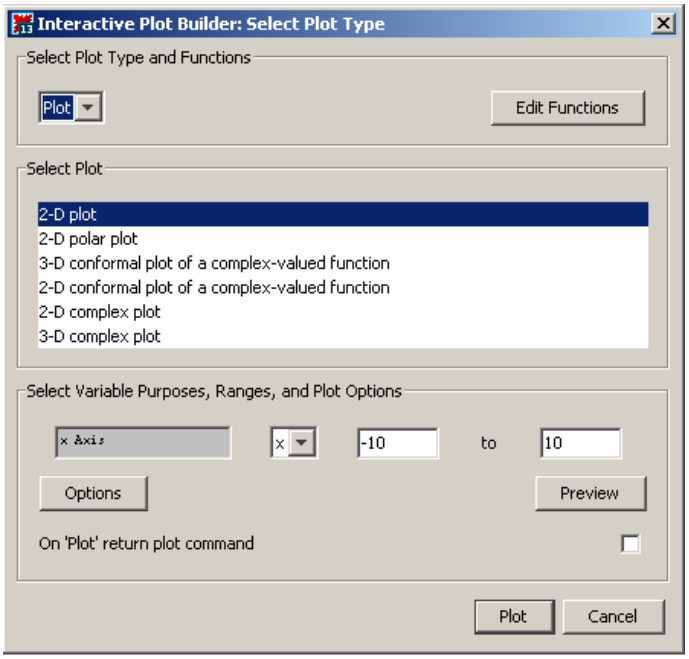
kliknout na něj pravým tlačítkem myši a z kontextové nabídky zvolit **Plots** > **2-D Plot** (v případě funkcí jedné proměnné). Zvolením **Plots** > **3-D Plot** z kontextové nabídky spolu se specifikací proměnných zobrazíme funkce dvou proměnných.

Dále můžeme využít pomocník **Plot Builder**, a to dvěma způsoby. Buď opět zapíšeme do dokumentu výraz z funkčního předpisu, klikneme pravým tlačítkem myši a zvolíme **Plots** > **Plot Builder**, nebo zamíříme do hlavního menu a vybereme **Tools** > **Assistants** > **Plot Builder...** . V prvním případě se objeví okénko **Interactive Plot Builder** (obrázek 5.31), v němž upřesníme typ vykreslení. Pokud uvažujeme funkci jedné proměnné, obvyklou volbou je **2-D Plot**. Můžeme však volit i jiné jako například vykreslení v polárních souřadnicích (**2-D polar plot**). Obdobně je tomu u trojrozměrného vykreslování funkcí dvou proměnných. Kliknutím na tlačítko **Plot** graf zobrazíme v dokumentu.

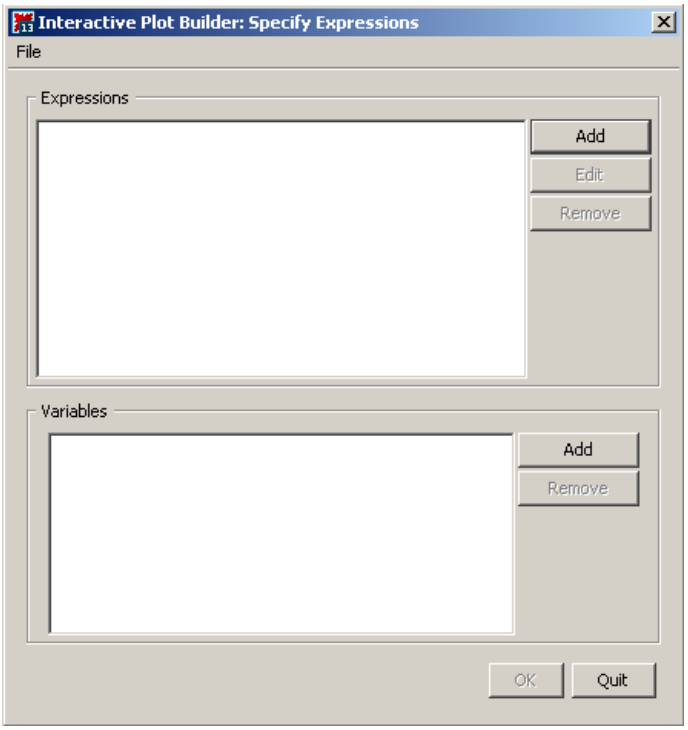
V druhém případě, kdy **Plot Builder** vyvoláme z hlavního menu, se nám objeví okénko (viz obrázek 5.32), do nějž zadáme výraz z předpisu funkce, kterou chceme zobrazit (zadání nám umožní tlačítka **Add**, resp. **Edit**), a proměnné (pokud výraz obsahuje pouze proměnné, systém je vyplní sám). Kliknutím na tlačítko **OK** přejdeme do již známého okénka pro zvolení typu vykreslení (obrázek 5.31).

Nakonec máme možnost k vykreslení grafu funkce použít příkaz **plot** v případě funkce jedné proměnné a příkaz **plot3d** pro funkci dvou proměnných. Příkazu **plot3d** musíme zadat i rozsahy hodnot, kterých mohou proměnné nabývat. Při trojrozměrném vykreslování se graf standardně zobrazuje bez souřadných os (obrázek 5.33). Oběma příkazům (**plot**, **plot3d**) můžeme na vykreslení zadávat jak funkce, tak výrazy.

Při vykreslování je k dispozici několik atributů měnících podobu grafu. Opět je několik možností, jak atributy zadávat. Při použití pomocníka **Plot Builder** se v okénku **Interactive Plot Builder** (obrázek 5.31) objevuje tlačítko **Options**. Kliknutím na toto tlačítko



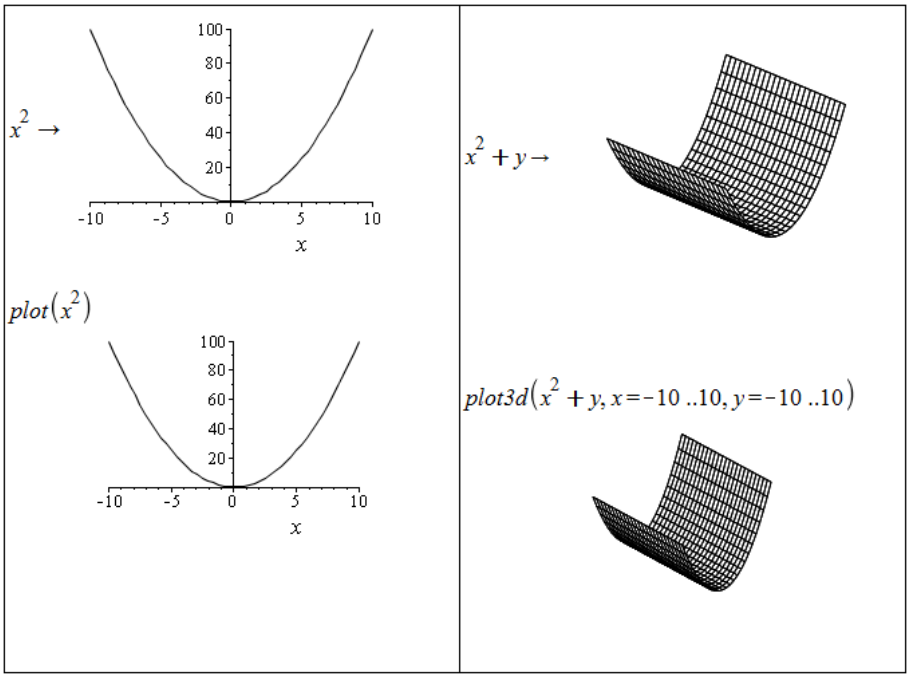
*Obrázek 5.31: Zvolení typu vykreslení v **Plot Builder**.*



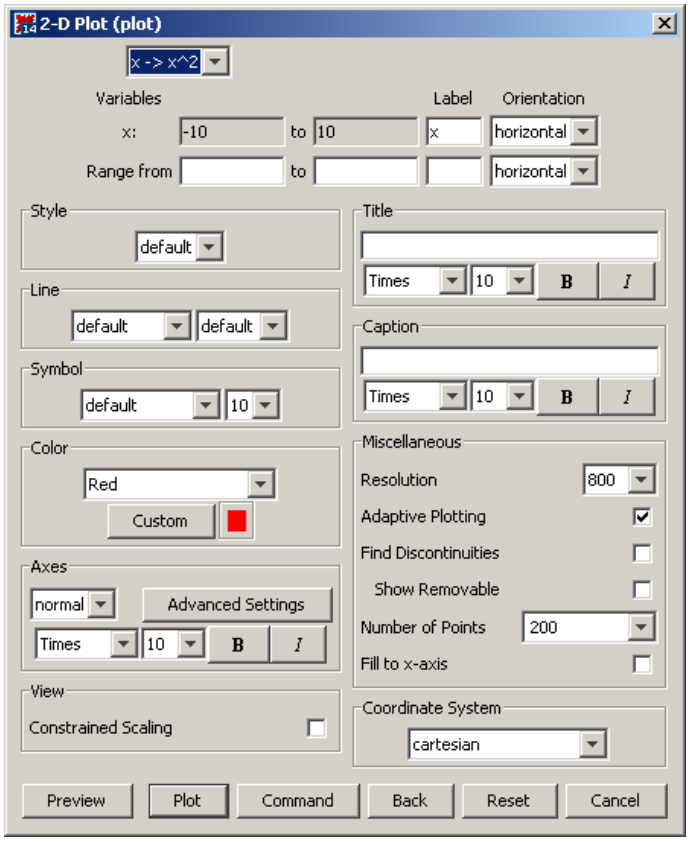
*Obrázek 5.32: Okénko pro zadání výrazu z předpisu funkce a proměnných v **Plot Builder**.*

přejdeme na okénko (viz obrázek 5.34) umožňující nastavit parametry vykreslení jako jsou rozsah hodnot závisle i nezávisle proměnné, barva a styl vykreslované křivky, titulek grafu, legenda atd. Užitečné je navíc tlačítko **Preview** umožňující předběžně si prohlédnout současný stav a následně pokračovat v dalším nastavování atributů vykreslení grafu.

Při použití příkazu **plot** (resp. **plot3d**) můžeme totéž provést specifikací nepovinných parametrů jako jsou **thickness** pro tloušťku křivky, **color** pro její barvu, **discont** pro zob-

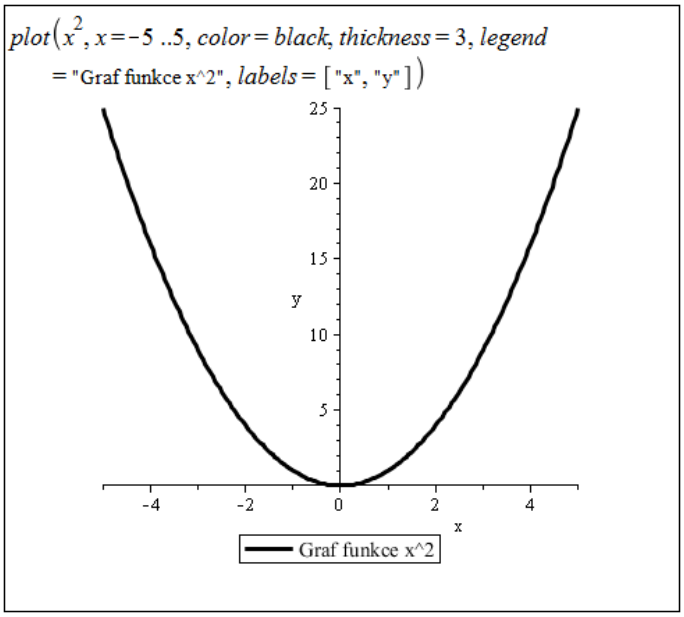


*Obrázek 5.33: Vykreslení grafů pomocí kontextové nabídky a příkazů **plot**, **plot3d**.*

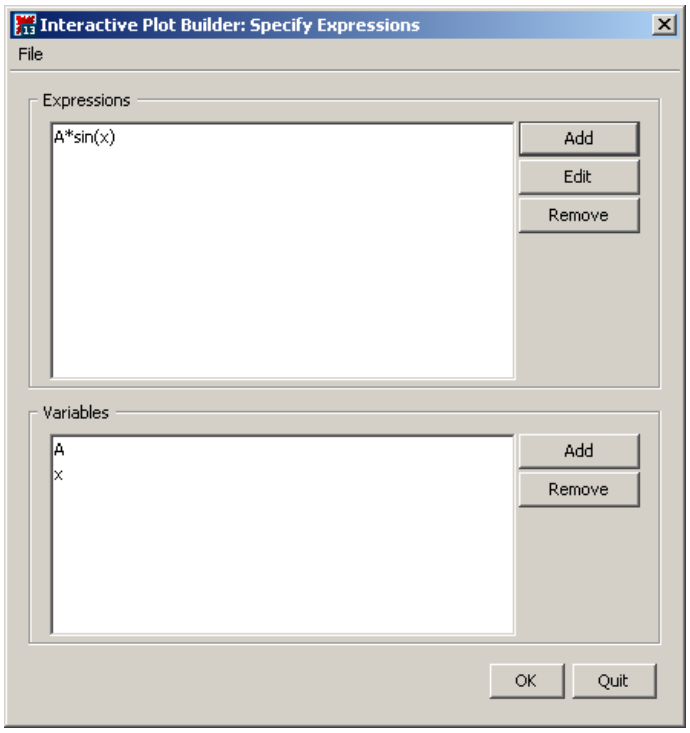


*Obrázek 5.34: Okénko **Plot Builder** pro nastavení parametrů grafu.*

razení nespojitostí, **labels** pro popisky os, **legend** pro tvar legendy u obrázku, **axes** pro nastavení souřadých os a další. Ukázku použití příkazu **plot** s nastavením některých nepovinných parametrů nabízí obrázek 5.35.



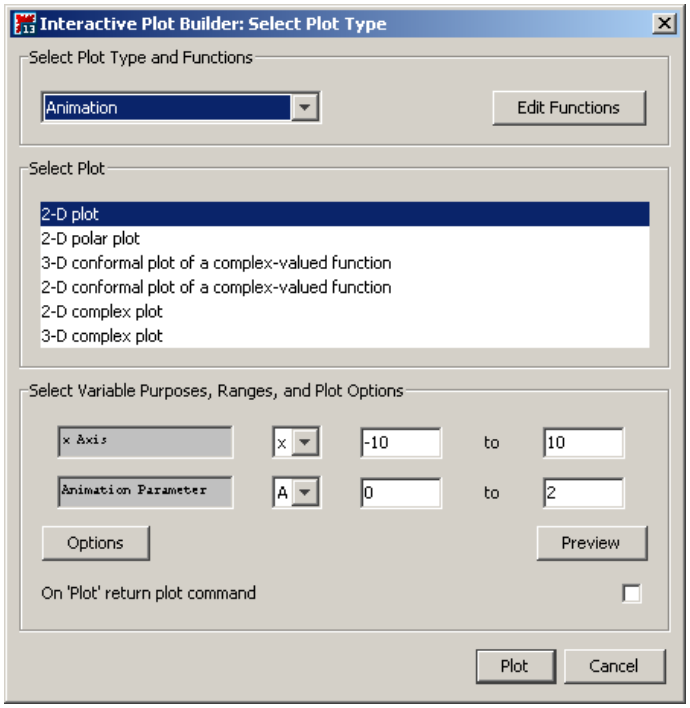
Obrázek 5.35: Vykreslení grafu při specifikaci některých nepovinných parametrů.



*Obrázek 5.36: Zadání výrazu z předpisu funkce v **Plot Builder**.*

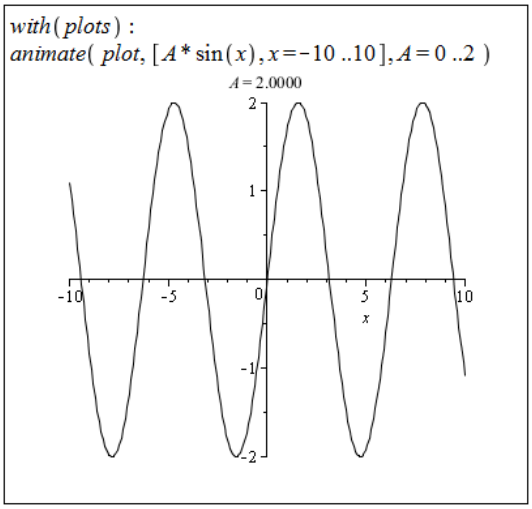
I u vytvořeného grafu lze upravovat jeho vzhled. Jednak můžeme na graf kliknout pravým tlačítkem myši a z kontextové nabídky vybírat vlastnosti grafu, které jsme mohli měnit již dříve, nebo můžeme využít kontextové lišty těsně nad dokumentem. Po kliknutí levým tlačítkem myši na graf se ve zmíněné liště zobrazí nástroje skupiny nazvané **Plot**. K dispozici je též skupina s názvem **Drawing**. Nástroje v těchto skupinách umožňují do hotového grafu přidávat text, kreslit, či jinak upravovat.

V systému Maple můžeme též vytvářet animace. Animace se skládá z několika grafů,



*Obrázek 5.37: Zvolení druhu vykreslení (animace) v **Plot Builder**.*

které jsou po spuštění zobrazené v sekvenci za sebou. Vytvoříme ji buď příkazem **animate** z balíku **plots**, nebo pomocí **Plot Builder**. Obrázky 5.36 a 5.37 ukazují nastavení **Plot Builder** pro vytvoření animace, obrázek 5.38 ilustruje tentýž příklad na použití příkazu **animate**.



*Obrázek 5.38: Animace vytvořená příkazem **animate**.*

Ukončení **Plot Builderu**, resp. provedení příkazu, umístí do dokumentu „prázdný“ graf. Kliknutím na něj zobrazíme skupinu nástrojů v kontextové liště s názvem **Animation**. Pomocí těchto nástrojů můžeme animaci spustit, změnit její rychlost, podívat se na libovolný snímek animace atd.

Animace můžeme upravovat stejně jako grafy (vytvořené příkazy **plots**, **plots3d**), tj. měnit tloušťku, barvu a druh křivky, souřadné osy, legendu apod. Navíc máme k dispozici

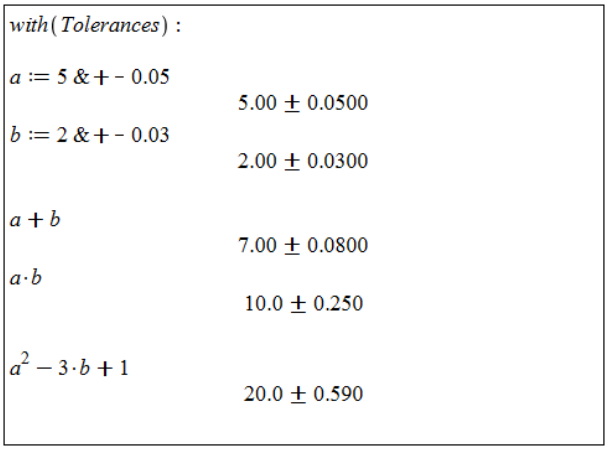
několik nepovinných parametrů, díky nimž můžeme například určit počet grafů, z nichž se animace skládá, nebo kolik grafů má zůstat trvale zobrazených (po spuštění animace).

5.7 Práce s neurčitostmi

Pro práci s neurčitostmi nabízí systém Maple několik nástrojů. Jejich použití závisí zpravidla na množství informace, kterou máme.

5.7.1 Intervalová aritmetika

V případě, že známe pouze velikost (intervalu) neurčitosti, využíváme intervalové aritmetiky implementované v balíku **Tolerances**. Proměnné (resp. konstanty) definujeme spolu s intervalem chyby (neurčitosti) pomocí symbolu **&+−** a pracujeme s nimi jako s „obyčejnými“ proměnnými (resp. konstantami), viz obrázek 5.39.



*Obrázek 5.39: Intervalová aritmetika s balíkem **Tolerances**.*

5.7.2 Scientific Error Analysis

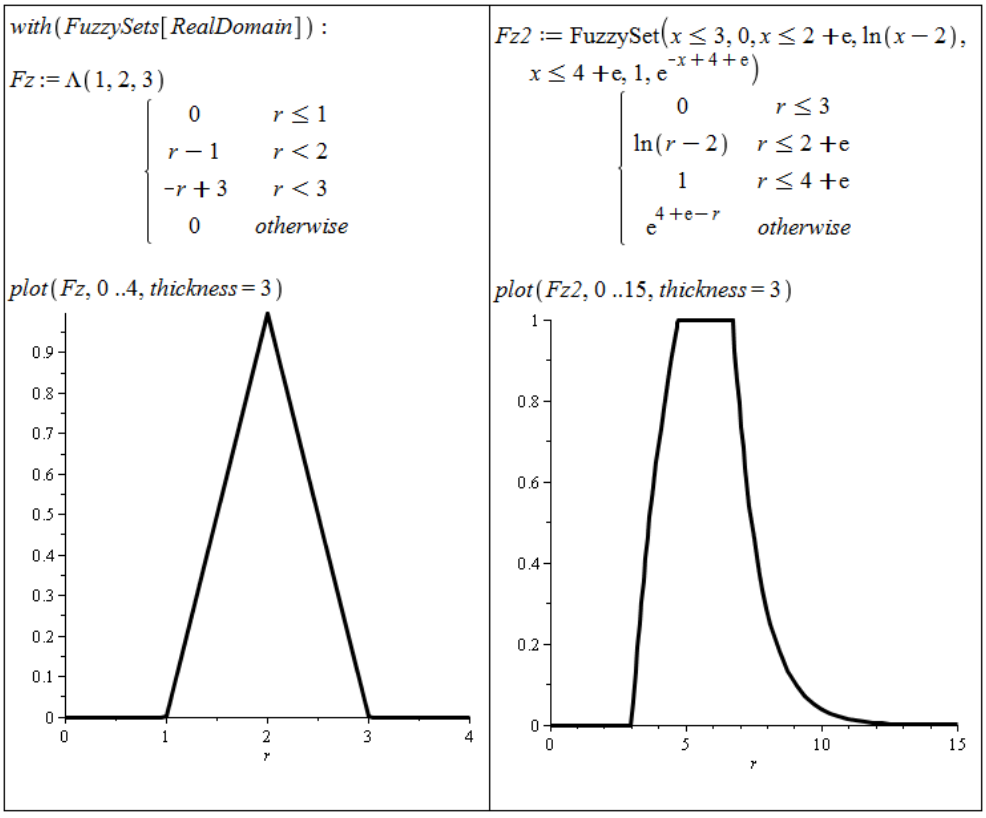
Balík **ScientificErrorAnalysis** umožňuje pracovat s veličinami, které mají nějakou střední hodnotu a přiřazenou neurčitost (resp. chybu). Na rozdíl od intervalové aritmetiky je nyní interval neurčitosti prokládán normálním rozložením hodnot se střední hodnotou ve středu intervalu a směrodatnou odchylkou rovnou délce poloviny intervalu. Navíc je možné přiřazovat veličinám korelaci (lineární závislost) mezi sebou. Veličiny definujeme příkazem **Quantity**, propagaci chyb daným výrazem určíme příkazem **combine**, korelaci nastavíme příkazem **SetCorrelation**. Ilustraci poskytuje obrázek 5.40.

5.7.3 Fuzzy Sets Toolbox

Toolbox **FuzzySets** není standardní součástí systému Maple, a je nutné jej dokoupit. Po instalaci funguje jako klasický balík příkazů, který umožňuje využívat fuzzy logiku a konstruovat a používat fuzzy množiny. Fuzzy množina může být definována obecně na libovolné množině objektů U, je však možné (a často vhodné) omezit balík **FuzzySets** pouze na

```
with(ScientificErrorAnalysis) :

a := Quantity(5, 0.05)
                    Quantity(5, 0.05)
b := Quantity(2, 0.03)
                    Quantity(2, 0.03)

combine(a + b, errors)
                    Quantity(7., 0.05830951895)
combine(a·b, errors)
                    Quantity(10., 0.1802775638)
combine(a^2 − 3·b + 1, errors)
                    Quantity(20., 0.5080354318)

SetCorrelation(a, b, 1)

combine(a + b, errors)
                    Quantity(7., 0.08000000000)
combine(a·b, errors)
                    Quantity(10., 0.2500000000)

SetCorrelation(a, b, −0.5)

combine(a + b, errors)
                    Quantity(7., 0.04358898944)
combine(a·b, errors)
                    Quantity(10., 0.1322875656)
```

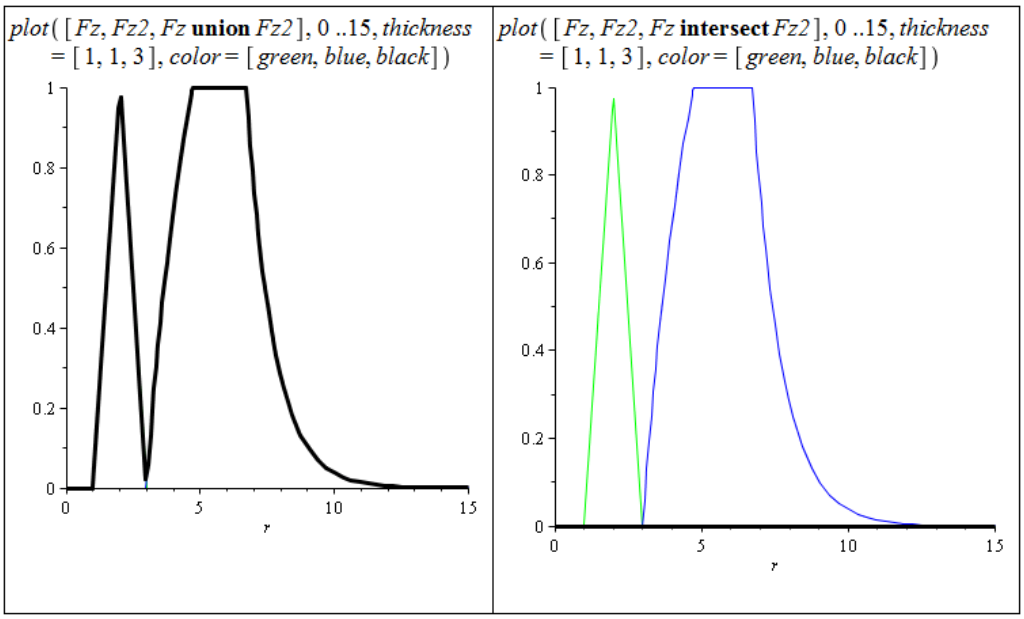
Obrázek 5.40: Práce s balíkem ***ScientificErrorAnalysis****.*



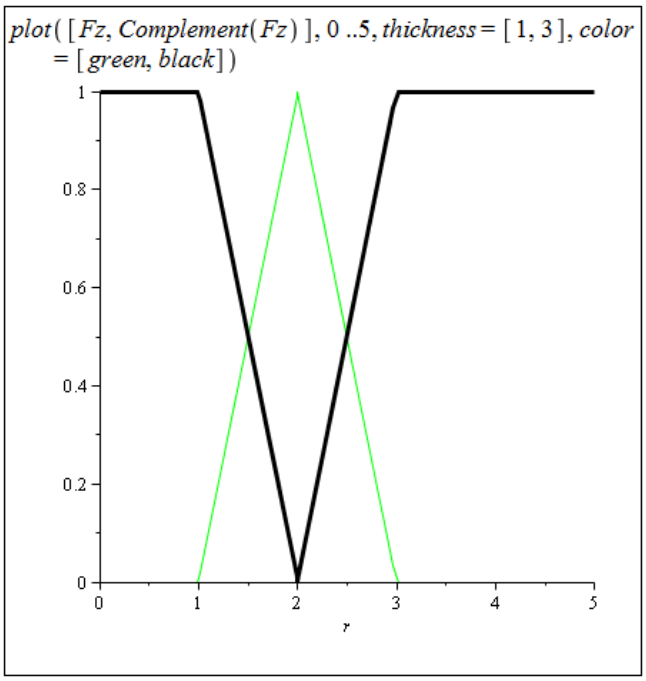
Obrázek 5.41: Konstrukce fuzzy množiny a její zobrazení.

množinu reálných čísel. K dispozici je několik konstruktorů fuzzy množin, přičemž ten nejuniverzálnější je téměř totožný s příkazem **piecewise** pro tvorbu po částech definovaných funkcí.

Obrázek 5.41 ilustruje načtení balíku **FuzzySets** s omezením na množinu reálných čísel a konstrukci dvou různých fuzzy množin s následným vykreslením jejich funkcí příslušnosti. V prvním případě (vlevo) je použit konstruktor Λ vytvářející fuzzy množinu s funkcí příslušnosti, jejíž tvar odpovídá tomuto řeckému písmenu. Napravo je fuzzy množina vytvořena pomocí obecného konstruktoru (který je analogický k příkazu **piecewise**).



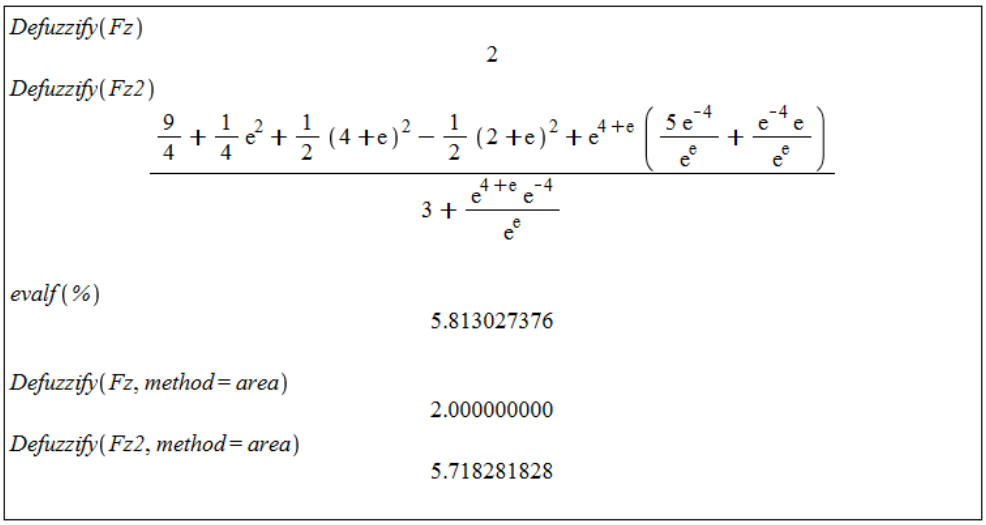
Obrázek 5.42: Zobrazení sjednocení a průniku fuzzy množin.



Obrázek 5.43: Komplement fuzzy množiny.

S fuzzy množinami můžeme provádět množinové operace. Na obrázku 5.42 jsou vyznačeny operace sjednocení (vlevo) a průnik (vpravo) právě vytvořených množin. K dispozici přitom máme i další množinové operace jako jsou odečítání množin, komplement množiny (v univerzu) a další. Ukázku komplementu fuzzy množiny představuje obrázek 5.43. Na obrázcích 5.42, 5.43 jsou vždy vykresleny funkce příslušnosti původních množin i těch výsledných (po aplikaci dané operace). Výsledné množiny jsou znázorněny tlustou čarou.

Balík **FuzzySets** poskytuje též prostředky tzv. *defuzzifikace*, tedy přiřazení číselné hodnoty dané fuzzy množině. Standardní proces defuzzifikace přiřadí fuzzy množině její těžiště (první moment), je však možné použít i jiné typy defuzzyfikací. Pro omezení se na doménu reálných čísel je k dispozici ještě metoda hledající medián funkce příslušnosti (specifikuje se pomocí atributu **method=area**), viz obrázek 5.44.



Obrázek 5.44: Defuzzifikace.

5.7.4 Balík Statistics

Pro práci s nástroji matematické statistiky a analýzy dat je určen balík **Statistics**. Nabízí širokou škálu příkazů, mezi nimiž jsou i příkazy pro vytváření náhodných veličin mnoha různých pravděpodobnostních rozložení a práci s nimi. Z pohledu neurčitostí se zaměřme právě na generování náhodných veličin, jež využíváme při Monte Carlo simulacích. Na obrázku 5.45 můžeme vidět načtení balíku **Statistics** a balíku **plots**, který potřebujeme k použití příkazu **display**. Definujeme náhodnou veličinu s lognormálním rozložením pravděpodobnosti (se střední hodnotou 1340 a směrodatnou odchylkou rovnou 0.6) a vykreslíme její hustotu. Následně vygerujeme 10 realizací této veličiny, kterým přiřadíme příkazem **Rank** pořadí (od nejmenší hodnoty). Poté vygenerujeme 1000 realizací definované náhodné veličiny, opět vykreslíme její hustotu, tentokrát spolu s histogramem vygenerovaných hodnot. Příkazy **Mean** a **Variance** určíme střední hodnotu a rozptyl vygenerovaných hodnot.

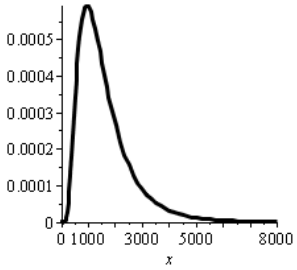
5.8 Základy programování

Doposud jsme vykonávali příkazy jednotlivě. Systém Maple obsahuje též programovací jazyk, který umožňuje vytváření složitějších programových konstrukcí.

with(*Statistics*) :
with(*plots*) :

Definice náhodné veličiny
Th := *RandomVariable*(*LogNormal*(ln(1340), 0.6))
_*R*

Vykreslení pravděpodobnostního rozložení hodnot
plot(*PDF*(*Th*, *x*), *x* = 0 ..8000, *thickness* = 3)



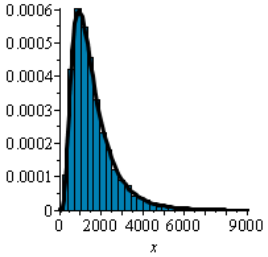
Vygenerování 10 realizací definované náhodné veličiny
Th_sample := *Sample*(*Th*, 10)
[1291.74658, 1033.77447, 937.85978, 546.49109, 986.79805,
1531.45303, 2846.89159, 535.20094, 723.81625,
1298.74874]

Pořadí vygenerovaných hodnot
Rank(%)
[7 6 4 2 5 9 10 1 3 8]

Vygenerování 10000 realizací definované náhodné veličiny
Th_sample := *Sample*(*Th*, 10000)

$$\begin{bmatrix} 1 .. 10000\ Vector_{row} \\ Data\ Type: float_8 \\ Storage: rectangular \\ Order: Fortran\_order \end{bmatrix}$$

Pravděpodobnostní rozložení spolu s histogramem
display([*plot*(*PDF*(*Th*, *x*), *x* = 0 ..8000, *thickness* = 3),
Histogram(*Th_sample*)])



Střední hodnota
Mean(*Th_sample*)
1595.035759

Rozptyl
Variance(*Th_sample*)
$1.094681903\ 10^6$

*Obrázek 5.45: Práce s balíkem **Statistics**.*

5.8.1 Podmíněný příkaz *if*

Syntax:

```
if      podminka1   then   prikazy1
elif    podminka2   then   prikazy2
elif    podminka3   then   prikazy3
          ⋮
                     else   prikazyN
end if
```

Jak můžeme vidět na obrázku 5.46, v podmíněném příkazu nemusí být použity všechny prvky uvedené výše. Podmínku tvoří vždy výraz, u nějž je možné rozhodnout, zda je pravdivý, či nikoliv. Příkazů, které se mají provést při splnění podmínky, může být více. Od sebe je oddělíme středníkem, případně dvojtečkou.

Při vytváření příkazu na více řádků použijeme k přechodu mezi řádky současného stisknutí kláves **Shift** a **Enter**. Stisk samotné klávesy **Enter** by způsobil vyhodnocení rozepsaného příkazu.

```
A := 3
                                3

if A < 5 then print("A je menší než pět.") end if
                        "A je menší než pět."

A := 6
                                6

if A < 5 then print("A je menší než pět.")
          else print("A je větší nebo rovno pěti.")
end if
                        "A je větší nebo rovno pěti."

if A = 6 then B := 7; C := 8; E := 156.89 end if
                                7
                                8
                             156.89
```

```
A := 3; B := 5
                                3
                                5

if (A < 5 and B < 5) then print("A i B jsou menší než pět.")

elif (A < 5 and B ≥ 5)
   then print("A je menší než pět, B je větší nebo rovno pěti.")

elif (A ≥ 5 and B < 5)
   then print("A je menší než pět, B je větší nebo rovno pěti.")

  else print("A i B jsou větší nebo rovny pěti.")

end if
          "A je menší než pět, B je větší nebo rovno pěti"
```

*Obrázek 5.46: Použití podmíněného příkazu **if**.*

5.8.2 Cyklus *for*

Syntax 1:

`for` *iterator* `from` *pocatek* `by` *prirustek* `to` *konec* `do` *prikazy* `end do`

Syntax 2:

`for` *promenna* `in` *vyraz* `do` *prikazy* `end do`

```
for i from 1 to 5 do print(i^2) end do
                                1
                                4
                                9
                               16
                               25

for i from 1 by 3 to 10 do print(i^2) end do
                                1
                               16
                               49
                              100
```

```
for i from 10 by -1 to 3 do
   if isprime(i) then print(i)
   end if
end do
                                7
                                5
                                3

seznam := [22, 97, 222, 397, 622]
                  [22, 97, 222, 397, 622]

for i in seznam do print(√(i + 3)) end do
                                5
                               10
                               15
                               20
                               25
```

*Obrázek 5.47: Použití cyklu **for**.*

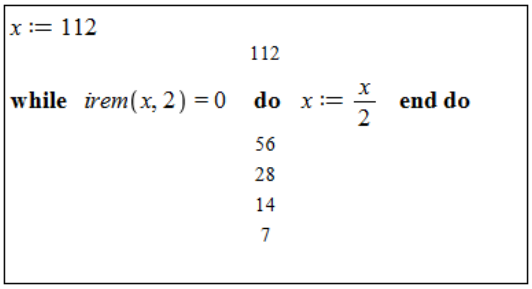
Na příkladech (obrázek 5.47) opět vidíme, že nemusíme využít všech prvků obecné syntaxe příkazu. Zejména pokud neuvedeme *prirustek*, bude system pracovat s hodnotou prirustku rovnou jedne. Příkazů, které se mají v jednom průchodu cyklem provést, může být opět více (a musí být od sebe odděleny středníkem, či dvojtečkou).

5.8.3 Cyklus *while*

Syntax:

`while` *podminka* `do` *prikazy* `end do`

Na obrázku 5.48 je pomocí cyklu **while** číslo 112 celočíselně dělené dvěma, dokud je zbytek po dělení rovný nule (zbytek po dělení určíme příkazem **irem**).



*Obrázek 5.48: Použití cyklu **while**.*

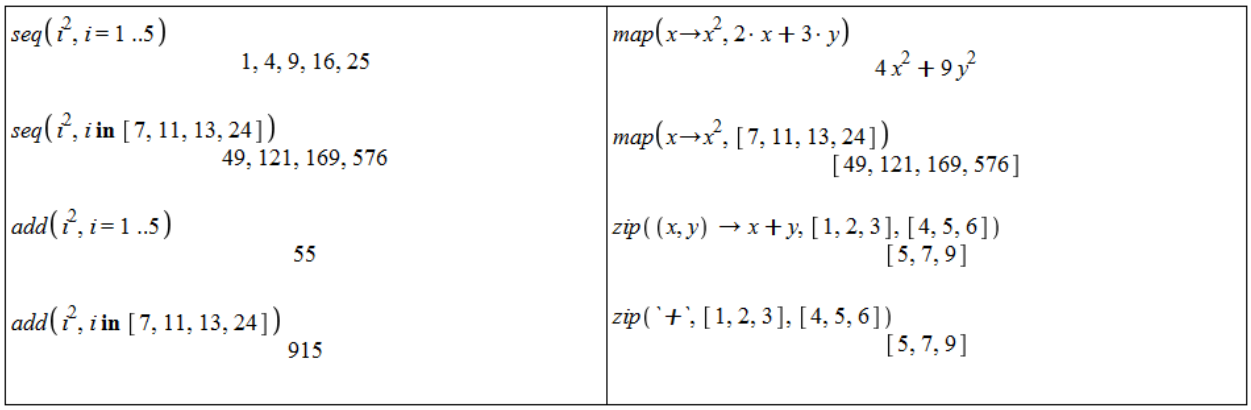
5.8.4 Iterativní příkazy

Maple obsahuje několik iterativních příkazů, které jsou optimalizovány pro určité specifické operace. Jejich přehled nabízí tabulka 5.2.

Tabulka 5.2: Přehled iterativních příkazů

Příkaz	**Funkce**
seq	vytvoří posloupnost
add	vypočítá numerický součet
mul	vypočítá numerický součin
select	vybere operandy, které vyhovují podmínce
remove	vybere operandy, které nevyhovují podmínce
selectremove	vypíše zvlášť operandy, které vyhovují podmínce, a operandy, které ji nevyhovují
map	aplikuje operaci na operandy výrazu
zip	aplikuje binární operaci na operandy dvou seznamů či vektorů

Obrázek 5.49 ilustruje použití příkazů **seq**, **add**, **map** a **zip** na jednoduchých příkladech.



Obrázek 5.49: Použití iterativních příkazů.

Literatura

[1] Adam, J.A.: Mathematics in Nature. Princeton and Oxford (2003)

[2] Barnes, B., Fulford, G.R.: Mathematical Modelling With Case Studies: A Differential Equation Approach Using Maple. CRC Press, London (2002)

[3] Begon, M., Harper, J.L., Townsend, C.R.: Ecology: individuals, populations and communities. Blackwell Scientific Publications, Oxford (1990)

[4] Ellner, S.P., Guckenheimer, J.: Dynamic Models in Biology. Princeton University Press, Princeton, New Jersey (2006)

[5] Elton, C.: The Ecology of Invasion by Animals and Plants. Methuen, London (1958)

[6] Eriksson, O.: Sensitivity and Uncertainty Analysis Methods, with Applications to a Road Traffic Emission Model. Dissertation thesis, Department of Mathematics, Linköping University (2007)

[7] Gander, W., Hřebíček, J.: Solving Problems in Scientific Computing Using Maple and MATLAB. 4th, expanded and rev. ed. Springer, Heidelberg (2004)

[8] Gause, G.F.: The struggle for existence. Williams and Wilkins, Baltimore (1934)

[9] Holčík, J., Fojt, O.: Modelování biologických systémů (Vybrané kapitoly). ÚBMI FEI VUT, Brno (2001)

[10] Holling, C.S.: The functional response of predator to prey density and its role in mimicry and population regulation. Mem. Entomol. Soc. Canada, vol. 45, pp. 1-60 (1965)

[11] Hřebíček, J., Hejč, M., Holoubek, I., Pešl, J.: Current Trends in Environmental Modelling with Uncertainties. In Proceedings of the iEMSs Third Biennial Meeting "Summit on Environmental Modelling & Software". Burlington : International Environmental Modelling and Software Society, pp. 342-347 (2006)

[12] Hřebíček, J., Žižka, J.: Vědecké výpočty v biologii a biomedicíně [online]. [cit 2010-07-29]. Dostupný z WWW: <`http://portal.med.muni.cz/download.php?fid=502`>

[13] Hřebíček, J., Škrdla, M.: Úvod do matematického modelování. Masarykova univerzita, Brno (2006)

[14] Isukapalli, S.: Uncertainty analysis of transport transformation models. Ph.D. thesis, State University of New Jersey, New Brunswick, New Jersey (1999)

[15] Kalas, J., Pospíšil, Z.: Spojité modely v biologii. Masarykova univerzita, Brno (2001)

[16] Kalas, J., Ráb, M.: Obyčejné diferenciální rovnice. Masarykova univerzita, Brno (2001)

[17] Kobza, A.: Diferenční rovnice ve středoškolské matematice. Diplomová práce, Masarykova univerzita, Brno (2001)

[18] Kulhavy, J.: Biotické interakce [online]. [cit 2010-07-29]. Dostupný z WWW: <http://www.uel.cz/download/Prednaska_6_Biotricke%20interakce.pdf>

[19] Lepš, J.: Růst populace [online]. [cit 2010-07-29]. Dostupný z WWW: <http://botanika.bf.jcu.cz/suspa/vyuka/materialy/Rustpopulace.ppt>

[20] Lepš, J.: Kompetice [online]. [cit 2010-07-29]. Dostupný z WWW: <http://botanika.bf.jcu.cz/suspa/vyuka/materialy/Kompeticeprvacka.ppt>

[21] Leslie, P.H.: Some further notes on the use of matrices in population mathematicics. Biometrika, vol. 35, pp. 213-245 (1948)

[22] MacArthur, R.H.: Fluctuations of animal populations and a measure of community stability. *Ecology,* vol. 36, pp. 533–536 (1955)

[23] Madzia, L.: Zajíc polní [online]. [cit 2010-07-29]. Dostupný z WWW: <http://www.prirodainfo.cz/karta.php?cislo=3080.00>

[24] Novák, V.: Základy fuzzy modelování. BEN − technická literatura, Praha (2000)

[25] Odum, E.P.: Fundamentals of Ecology. W.B. Saunders Company, Philadelphia (1971). Český překlad: Základy ekologie. Academia, Praha (1977)

[26] Pešl, J.: Uncertainty handling in the environmental modeling using computer algebra systems with online data manipulation. Ph.D. thesis, Masaryk University, Brno (2005)

[27] Pospíšil, Z.: Dynamika populací s oddělenými generacemi [online]. [cit 2010-07-29]. Dostupný z WWW: <http://www.math.muni.cz/~pospisil/FILES/DPNG.pdf>

[28] Saltelli, A., Chan, K., Scott, E.M.: Sensitivity Analysis. John Wiley and Sons, Ltd.: West Sussex, England (2000)

[29] Saltelli, A.: Global Sensitivity Analysis: An Introduction. In Proc. 4th International Conference on Sensitivity Analysis of Model Output, pp. 27 – 43 (2004)

[30] Sharov, A.: Quantitative Population Ecology. On-Line Lectures [online]. [cit 2010-07-29]. Dostupný z WWW: <http://home.comcast.net/~sharov/PopEcol/>

[31] The Council for Regulatory Environmental Modeling.: Draft Guidance on the Development, Evaluation, and Application of Regulatory Environmental Models [online]. [cit 2010-07-29]. Dostupný z WWW: <http://www.modeling.uga.edu/tauc/other_papers/CREM%20Guidance%20Draft%2012_03.pdf>

[32] Urbánek, J.: Numerická stabilita v environmentálním modelování s neurčitostmi. Diplomová práce, Masarykova univerzita, Brno (2009)

[33] Yeargers, E.K., Shonkwiler, R.W., Herod, J.V.: An Introduction to the Mathematics of Biology. With Computer Algebra Models. Birkhauser, Boston-Basel-Berlin (1996)

Obsah

Summary

This book entitled *Introduction to Mathematical Modelling Using Maple* is primarily intended for students and teachers of the study program Computational Biology at the Faculty of Science, Masaryk University; however, it can be used by anyone interested in mathematical modelling. The book provides a short theoretical basis in the field of mathematical modelling, as well as several solved practical examples using the modern information and communication technology (ICT) tool Maple.

The first chapter introduces mathematical modelling. It gives the definition of a mathematical model, reveals the parts of the model, describes model types, shortly presents ICT tools for work with mathematical models, and also mentions the notion of uncertainty in mathematical models.

The second chapter presents methodology of mathematical modelling. It summarizes the general principles in the mathematical modelling and characterizes the mathematical modelling concept using ICT. The steps of creating the model are also mentioned here.

The third chapter illustrates several models from the field of population dynamics. The first basic model of unlimited population growth is shown in detail at each step of the model development. Then the limited population growth model and the population growth under the predator pressure model follow as modifications of one population growth model. Further, the models of interacting populations are presented. Competition models, Symbiosis models, Predation models, or the classical predator-prey models of the Leslie type or Gausse type are illustrated for the models of two interacting populations. The community of three or more species model is shown in several examples at the end of the chapter. Each example in this chapter is accompanied by a graphical illustration and by a Maple source code solving the example. All the models are presented in both discrete and continuous form; however, for the illustrations and computations of model of more than one population growth only the continuous case is used.

The fourth chapter analyses data uncertainties of the unlimited population growth model by means of three different methods: interval arithmetic, fuzzy sets, and probability analysis. The sensitivity analysis of three models (unlimited population growth model, limited population growth model, and population growth under the predator pressure model) is performed here as well.

The final chapter is focused on a short introduction to Maple, release 14. This version of Maple is also used here in the basic manipulation as well as in more complex tasks when dealing with mathematical models.

Úvod do matematického modelování s využitím Maple
prof. RNDr. Jiří Hřebíček, CSc., doc. RNDr. Zdeněk Pospíšil, Dr.,
Mgr. et Mgr. Jaroslav Urbánek

Recenzenti: doc. RNDr. Stanislav Bartoň, CSc., prof. RNDr. Ivanka Horová, CSc.
Obálka: Radim Šustr, DiS., ve spolupráci s doc. PaedDr. Daliborem Martiškem, Ph.D.
Sazba: Mgr. et Mgr. Jaroslav Urbánek
Vydalo: AKADEMICKÉ NAKLADATELSTVÍ CERM, s.r.o. Brno
Purkyňova 95a, 612 00 Brno
www.cerm.cz
Tisk: FINAL TISK s.r.o. Olomučany
Náklad: 200 ks
Vydání: první
Vyšlo v roce 2010

ISBN 978-80-7204-691-1