LISP-STAT: PROSTŘEDÍ PRO STATISTICKÉ VÝPOČTY A GRAFIKU

Ivan KŘIVÝ
OU PřF, KM

Abstract: Basic instructions of XLISP-STAT (an implementation of the Lisp-Stat environment for statistical computing and dynamic graphics) are briefly described. The XLISP-STAT environment is found to be modern (object–oriented), flexible (open and extensible) and easily attainable for everybody. An example is given to illustrate the possibilities of using the XLISP-STAT in exploratory data analysis.

Резюме: В работе изложены основные инструкции системы XLISP-STAT (имплементации среды Lisp-Stat для статистических вычислений и динамической графики). Показано, что эта среда представляет объекто–ориентированное, открытое (способное расширения) и легко доступное средство для разведочного анализа данных. Возможности применения XLISP-STAT иллюстрированы на примере из области линейной регрессии.

1 Úvod

S rozvojem výpočetní techniky se přirozeně zvyšují nároky na statistický software, jeho rychlost a kvalitu, vypovídací schopnost získaných výsledků a uživatelský komfort. Úsilí odborníků směřuje k navržení statistického výpočetního prostředí, které

- využívá interaktivního programovacího jazyka vysoké úrovně
- respektuje principy objektově orientovaného programování při reprezentaci datových struktur a statistických modelů
- poskytuje prostředky pro grafické programování (dynamické grafy)
- umožňuje zpracovávat i nestandardní statistická data

V podstatě existují dva odlišné přístupy k vytvoření takového prostředí:

1. vytvoření nového specializovaného statistického jazyka (S-PLUS)
2. adaptace již existujícího moderního jazyka pro statistické výpočty

Lisp-Stat [5] představuje výsledek druhého z uvedených přístupů, vycházejícího z jazyku Lisp. Zahrnuje v sobě interaktivní jazyk Lisp, funkce pro základní statistické výpočty, objektově orientovaný systém na podporu grafického programování, prostředky pro konstrukci grafů a interface na Windows. Poskytuje i řadu dalších výhod:

- bezplatná distribuce software
- možnost definovat nové funkce, objekty a jejich metody
- velmi dobrá a průběžně doplňovaná dokumentace

Zatím převládá implementace XLISP-STAT založená na dialektu XLISP, jež pracuje jako aplikace pod Windows (MS 3.1 Windows pro DOS na IBM PC nebo X-Windows pro UNIX na pracovních stanicích Sun). Existuje také implementace XLISP-STAT pro personální počítače MacIntosh.

2 Charakteristika Lispu

Veškerá interakce s prostředím Lisp-Stat se děje prostřednictvím konverzace mezi uživatelem a interpretem Lispu. Uživatel zadává výrazy, interpret je vyhodnocuje a vrací jejich hodnoty. Složitější výrazy se zapisují v tzv. *prefixové notaci* , tj. ve tvaru

$(< operator > < operand - 1 > ... < operand - n >)$

Tyto výrazy se zpravidla vyhodnocují jako aplikace funkce $< operator >$ na argumenty $< operand - 1 >$, ... , $< operand - n >$.

Data jsou v Lispu dvojího typu: jednoduchá a složená. Za jednoduchá data se považují:

- čísla (celá, reálná, komplexní)
- logické konstanty (**t** a **nil**)
- znaky (např. #\a)
- řetězce (např. "abc...z")
- symboly

Symboly jsou reálná fyzická data uložená v paměti. Daný symbol může reprezentovat jak proměnnou, tak i funkci.

Základním nástrojem pro konstrukci složených dat jsou seznamy (**lists**) a funkční aparát na jejich vytváření a zpracování. Přestože mají sekvenční strukturu, mohou být použity i k reprezentaci neuspořádaných souborů položek (např. množin).

Pro konstrukci seznamů se používá výrazů (konstrukcí):

(list $< item - 1 > ... < item - n >$**)**
(quote $(< item - 1 > ... < item - n >)$**)**
'$(< item - 1 > ... < item - n >)$

Položkou seznamu může být libovolný datový objekt, tedy i seznam. *Položky seznamu se číslují zásadně od nuly.* K reprezentaci numerických dat je možno použít i datových typů **vector** a **array**.

Proměnným (Lispu) se přiřazují hodnoty pomocí výrazu

(def < *name* > < *value* >**)**

Např. zápis

(def x (list 10 15 20 25))

definuje proměnnou **x** jako seznam tvořený čtyřmi uvedenými numerickými položkami.

Základní operací při programování v Lispu je definice *funkce*. Taková definice má tvar

(defun < *name* > < *parameters* > < *body* >**)**

v němž < *name* > značí jméno funkce, < *parameters* > seznam jejich formálních parametrů (argumentů) a < *body* > jeden nebo více výrazů. Např. funkce pro výpočet součtu čtverců hodnot proměnné **x**, jež reprezentuje jednoduchý seznam, se definuje takto:

(defun sum-of-squares (x) (sum (* x x)))

Většina funkcí má pevný počet argumentů, některé z nich však mohou být volitelné. V případě volitelných argumentů lze zadat i „default“ hodnoty.

V Lispu existuje celá řada vestavěných funkcí a speciálních forem pro:

- zajištění numerických výpočtů, např. funkce **+, -, *, /, log, exp, sqrt**
- vyhodnocení logických výrazů, např. predikáty **<, =, >** a formy **and, or, not**
- konstrukci a zpracování seznamů, např. funkce **list, append, member, length**
- řízení vstupů a výstupů, např. funkce **print, format**

Lisp jako jazyk vysoké úrovně přirozeně disponuje prostředky umožňujícími:

- vyhodnocovat podmíněné výrazy, např. pomocí **if, cond**
- programovat cykly, např. pomocí **dotimes, do**
- definovat nejen globální, ale i lokální proměnné a funkce, např. pomocí **let**
- využívat knihoven funkcí definovaných v různých souborech (modulech)

3 Výpočty v prostředí Lisp-Stat

Prostředí Lisp-Stat nabízí uživateli veškeré prostředky, kterými disponuje Lisp.

3.1 Čtení dat

V případě malého rozsahu dat lze proměnným přiřadit hodnotu pomocí funkce **list** nebo speciální formy **quote**.

Rozsáhlá data jsou zpravidla uložena v nějakém souboru. Jsou-li uspořádána ve sloupcích, postupuje se takto:

1. Celý soubor se načte a uloží, např. do proměnné **mydata**, pomocí

   (def mydata (read-data-columns "mydata.txt" $< k >$**))**

   kde $< k >$ značí počet datových sloupců. Pokud není hodnota $< k >$ uvedena, systém ji nastaví podle počtu položek na prvním řádku. Proměnná **mydata** reprezentuje seznam tvořený k dílčími seznamy.

2. Jednotlivým proměnným se přiřadí hodnoty pomocí funkce **select**

   (def x1 (select mydata 0))
   . . .
   (def xk (select mydata $< k - 1 >$**))**

Pokud nejsou data ve vstupním souboru uspořádána ve sloupcích, použije se funkce **read-data-file**, tedy

(def mydata (read-data-file "mydata.txt"))

V tomto případě proměnná **mydata** představuje jednoduchý seznam (načtený po řádcích).

3.2 Systematická data

Taková data se generují pomocí funkcí **iseq, rseq** a **repeat**. Výraz

(iseq $< n > < m >$**)**

generuje seznam po sobě jdoucích celých čísel od $< n >$ do $< m >$. Analogický výraz

(rseq $< a > < b > < n >$**)**

vytváří seznam $< n >$ ekvidistantních reálných čísel v intervalu od $< a >$ do $< b >$ (včetně obou krajních hodnot).

Volání funkce **repeat** má obecně tvar

(repeat $<$ *list* $>$ $<$ *pattern* $>$**)**

kde $<$ *list* $>$ je seznam a $<$ *pattern* $>$ kladné celé číslo nebo seznam takových čísel (stejné délky jako $<$ *list* $>$). Tato funkce generuje data, jejichž položky se systematicky opakují. Např.

(repeat (list 1 2 3) 2) $\Rightarrow$ **(1 2 3 1 2 3)**
(repeat (list 1 2 3) (list 3 2 1)) $\Rightarrow$ **(1 1 1 2 2 3)**

Uvedená funkce je zvláště vhodná ke kódování úrovní sledovaných faktorů v analýze rozptylu.

3.3 Pseudonáhodná čísla

Pro generování pseudonáhodných čísel v Lisp-Statu slouží výrazy typu

($<$ *distribution* $>$**-rand** $<$ *N* $>$ $<$ *parameters* $>$**)**

které generují seznamy $<$ *N* $>$ pseudonáhodných čísel z následujících rozdělení
$<$ *distribution* $>$ (s příslušnými parametry):

- **uniform, normal, cauchy, gamma, beta, t, chisq, f, bivnorm**
- **binomial, poisson**

Uživatel má přirozeně možnost měnit násadu (**seed**) generátoru.

Např. výraz

(normal-rand 50)

generuje 50 pseudonáhodných čísel z rozdělení $N(0,1)$. Pokud uživatel potřebuje pseudonáhodná čísla z rozdělení např. $N(3,4)$, musí použít výrazu

(+ 3 (* 2 (normal-rand 50)))

Náhodný výběr z daného seznamu se realizuje pomocí funkce **sample**. Např. výraz

(sample (iseq 1 20) 10)

vytváří náhodný výběr o rozsahu 10 bez vracení ze seznamu (1 2 ... 20). V případě, že se vyžaduje výběr s vracením, je nutno navíc specifikovat hodnotu třetího (volitelného) argumentu, tedy

(sample (iseq 1 20) 10 t).

3.4 Distribuční funkce

Základní modul Lisp-Statu nabízí uživateli prostředky pro výpočet hodnot distribuční funkce, hustoty (pravděpodobnostní funkce) a kvantilů pro tytéž typy rozdělení jako v předcházejícím odstavci. Jde o funkce:

- < *distribution* >**-cdf** — distribuční funkce
- < *distribution* >**-dens** — hustota pravděpodobnosti
- < *distribution* >**-pmf** — pravděpodobnostní funkce
- < *distribution* >**-quant** — kvantily

Např.

(chisq-quant .975 3)

vrací hodnotu 97,5-procentního kvantilu rozdělení χ^2 se třemi stupni volnosti.

3.5 Operace se seznamy

V prostředí Lisp-Stat jsou některé funkce upraveny tak, že podporuji tzv. *vektorizovanou aritmetiku* , proto např.

(+ (list 1 2 3) 4) $\Rightarrow$ **(5 6 7)**

Lisp-Stat poskytuje speciální funkce pro práci s maticemi (např. **column-list, row-list, diagonal, print-matrix, matmult, determinant, inverse**) a funkci **solve** pro řešení soustavy lineárních algebraických rovnic.

Základní modul Lisp-Statu obsahuje řadu funkcí pro úpravu dat ve formě seznamů. Tyto funkce umožňují:

- vytvářet podmnožiny dat a vyřazovat některé údaje (**select** a **remove**)
- spojovat a rozpojovat data (např. **combine** a **split-list**)
- měnit hodnoty vybraných položek (**setf**)
- třídit data (**sort-data, rank** a **order**)
- provádět interpolaci a vyhlazování (např. **spline**)

Způsob použití těchto funkcí je zřejmý z následujících jednoduchých příkladů. Nechť jsou proměnné **x** a **y** definovány takto:

(def x (list 3 7 5 9 12 3))
(def y (list 1 4 0 8 2 11))

Pak

(select x 3) ⇒ **9**
(select x (which (< 3 x))) ⇒ **(7 5 9 12)**
(remove 3 x) ⇒ **(7 5 9 12)**
(combine x y) ⇒ **(3 7 5 9 12 3 1 4 0 8 2 11)**
(split-list x 3) ⇒ **((3 7 5)(9 12 3))**
(rozpojuje vychozi seznam na seznamy stejne delky)
(setf (select x 4) 10) ⇒ **10**
(vybira paty prvek seznamu a meni jeho hodnotu na 10)
(sort-data x) ⇒ **(3 3 5 7 9 12)**
(rank x) ⇒ **(0 3 2 4 5 1)**
(vraci seznam poradi prvku ve vychozim seznamu)
(order x) ⇒ **(0 5 2 1 3 4)**
(vraci seznam indexu nejmensiho, druheho nejmensiho,...,
nejvetsiho prvku ve vychozim seznamu)

3.6 Statistické funkce

Pro elementární statistické výpočty jsou k dispozici funkce: **min, max, sum, product, mean, standard-deviation, median, interquantile-range, cumsum, difference, pmin** a **pmax**. Způsob použití většiny z nich je zřejmý. Např.

(cumsum '(1 2 3 4)) ⇒ **(1 3 6 10)**
(difference '(1 3 6 10) ⇒ **(2 3 4)**
(pmin '(1 2 3 4) '(4 3 2 1)) ⇒ **(1 2 2 1)**
(pmax '(1 2 3 4) '(4 3 2 1)) ⇒ **(4 3 3 4)**

3.7 Poznámky k interpretu

Interpret Lisp-Statu poskytuje uživateli rozšířené služby, jež zahrnují:

- záznam konverzace s interpretem (funkce **dribble**)
- přístup ke třem posledním načteným výrazům, resp. jejich hodnotám (výrazy +, ++, +++, **resp. *, **, *****)
- úschovu hodnot proměnných a jejich zpětné načtení (funkce **savevar** a **load**)
- informace o funkcích, datových typech a některých proměnných (funkce **help** a **apropos**)

4 Grafické prostředky v Lisp-Statu

Součástí Lisp-Statu je objektově orientovaný systém navržený speciálně pro podporu interaktivní statistické práce. Objekt představuje zvláštní datovou strukturu, která obsahuje specifické informace o tomto objektu (atributy objektu) a navíc je schopna na požádání (zaslání zprávy) provádět určité akce (metody).

Zpráva se objektu zasílá pomocí funkce **send** ve tvaru

(send $<$ *object* $>$ $<$ *selector* $>$ $<$ $arg-1>$ ... $<$ $arg-n>$**)**

v němž $<$ *selector* $>$ je klíčový symbol identifikující danou zprávu a $<$ $arg-1>$, ..., $<$ $arg-n>$ příslušné argumenty zprávy.

Lisp-Stat zahrnuje řadu předdefinovaných prototypů objektů. Tyto prototypy pak slouží ke konstrukci jednotlivých instancí s konkrétním obsahem (objektů). Uživatel má samozřejmě k dispozici programové prostředky pro definování nových (vlastních) prototypů a jejich metod, jakož i pro generování instancí od všech definovaných prototypů.

Prototypy objektů v Lisp-Statu vytvářejí hierarchickou strukturu, ve které se respektuje princip dědičnosti. Nejvýše v této hierarchii stojí prototyp objektu, na nějž ukazuje hodnota globální proměnné ***object***.

4.1 Jednoduché grafy

Základní prostředky pro konstrukci jednoduchých grafů poskytuje prototyp **graph-proto**. Takové grafy tedy představují instance vytvořené od prototypu **graph-proto**. Složitější grafy (např. histogramy, bodové grafy) se odvozují od prototypů, které jsou ve srovnání s prototypem **graph-proto** specializovanější (mají bohatší obsah).

Pro grafickou reprezentaci jednorozměrných dat (uložených v proměnné **x**) slouží funkce **histogram** a **boxplot**. Výrazy

(histogram x) a **(boxplot x)**

generují příslušné grafy a umisťují je do nového grafického okna na monitoru.

Dvourozměrná data (uložená v proměnných **x** a **y**) lze zobrazit funkcemi **plot-points** a **plot-lines**.

(plot-points x y)	generuje bodovy graf (graf rozptylenosti)
(plot-lines x y)	generuje graf, jehoz body jsou spojeny useckami

Je-li definována funkce **f** jedné proměnné **x**, potom její graf v intervalu <**xl, xu**> se generuje výrazem

(plot-function #'f xl xu)

Např.

(plot-function #'sin (- pi) pi)

zobrazuje graf funkce **sin** v intervalu $< -\pi, \pi >$.

4.2 Dynamická grafika

Při studiu vztahů mezi třemi a více proměnnými jsou dosud uvedené prostředky nedostačující. Pro tento účel jsou k dispozici funkce **scatterplot-matrix** a **spin-plot**. Pomocí výrazu

(scatterplot-matrix (list x ... z))

vytvoříme matici, jejímiž prvky jsou bodové grafy pro jednotlivé páry specifikovaných proměnných. Všechny bodové grafy v matici jsou navzájem propojeny, což znamená, že vyznačíme-li jeden nebo více bodů v jednom grafu, automaticky se vyznačí tyto body i ve všech ostatních grafech. Výraz

(spin-plot (list x y z))

generuje trojrozměrný bodový graf s možností rotace kolem všech tří os. Graf funkce **f** dvou proměnných **x** a **y** na množině <**xl, xu**> × <**yl, yu**> vznikne zadáním výrazu

(spin-function #'f xl xu yl yu)

Takovým grafům (instancím prototypů **scatmat-proto**, resp. **spin-proto**) se říká *dynamické grafy* .

Všechny grafy je možno opatřit nadpisem i popisem jednotlivých os, a to přiřazením vhodných hodnot volitelným klíčovým argumentům **:title** a **:variable-labels**. Např.

(plot-points x y :title "Mygraph"
:variable-labels (list "varx" "vary"))

4.3 Mechanismus posílání zpráv

Vytvořené grafické objekty lze dotvářet i zásadně měnit mechanismem posílání zpráv. V takovém případě se nejprve objekt pojmenuje (např. pomocí funkce **def**) a teprve pak se vyžaduje provedení potřebných akcí.

U grafických objektů vytvořených podle všech standardních prototypů (včetně prototypu **graph-proto**) je možno zasíláním příslušných zpráv:

- určit počet proměnných, resp. experimentálních bodů (zprávy **:num-variables**, resp. **:num-points**)

- přidat nové experimentální body (**:add-points**)
- zadat nebo změnit pro každý bod jeho symbol, barvu a název (**:point-symbol, :point-color, :point-label**)
- zadat nebo změnit souřadnice libovolného bodu (**:point-coordinate**)
- zobrazit osy souřadnic, upravit počet značek na osách změnit rozsah zobrazování jednotlivých proměnných (**:x-axis, :y-axis, :range**)
- aplikovat lineární transformace (**:scale, :center, :transformation, :rotate-2**)
- zobrazit křivky ve formě lomených čar spojujících zadané body (**:add-lines**)

Objekty odvozené od specializovanějších prototypů mohou ovšem reagovat na více zpráv než objekty vytvořené podle prototypu **graph-proto**. Chce-li uživatel získat seznam všech zpráv, kterým daný objekt rozumí, pošle tomuto objektu zprávu s klíčovým slovem **:help**, tedy např.

(send fig :help)

Je-li uvažovaný objekt **fig** odvozen např. od prototypu **scatterplot-proto**, vydá interpret seznam více než 50 standardních zpráv, mezi nimiž je i zpráva **:abline** pro zakreslení přímky do bodového grafu. Podrobné informace o této zprávě (počet a typ argumentů, default hodnoty volitelných argumentů) se získají zasláním zprávy **:help** s klíčovým argumentem **:abline**, tj.

(send fig :help :abline)

Pak již může uživatel použít zmíněnou zprávu ve tvaru

(send fig :abline $< a >$ $< b >$**)**

kde $< a >$, $< b >$ jsou hodnoty parametrů ve směrnicovém vyjádření rovnice přímky ($y = a + bx$).

Všechny grafy odvozené od téhož prototypu znají stejné metody, seznamy metod pro objekty vytvořené podle různých prototypů se ovšem liší.

4.4 Pohyblivé obrázky

Zvláštním typem dynamických grafů jsou tzv. *pohyblivé obrázky* (**movie**). Jde o grafické objekty, které se v průběhu času systematicky mění (dynamická simulace). Uveďme jednoduchý příklad (viz [5]). Pomocí výrazu

(def h (histogram (normal-rand 20)))

vygenerujeme histogram pro výběr 20 pseudonáhodných čísel s rozdělením $N(0,1)$. Následné zadání výrazu

(dotimes (i 50)
(send h :clear :draw nil)
(pause 10)
(send h :add-points (normal-rand 20)))

způsobí, že se v grafickém okně postupně objevuje 50 histogramů, z nichž každý odpovídá nějakému výběru 20 pseudonáhodných čísel s rozdělením $N(0,1)$.

Vysvětlení.

dotimes je speciální forma pro konstrukci cyklu. Uvedený cyklus je tvořen třemi výrazy. Zpráva **:clear** s klíčovým argumentem **:draw** majícím hodnotu **nil** vymaže data, ale nekreslí. Zpráva **:add-points** přidává nová data a způsobí výstup nového grafu.

5 Regresní analýza v Lisp-Statu

5.1 Lineární regresní modely

Lineární regresní model se vytváří pomocí funkce **regression-model**. Výraz

(regression-model $< x > < y >$**)**

v němž $< x >$ je pro jednoduchou regresi seznam hodnot nezávisle proměnné a $< y >$ seznam hodnot závisle proměnné, vytvoří objekt (ne grafický) reprezentující příslušný model. V případě vícenásobné regrese má první argument funkce **regression-model** tvar **(list** $< x-1 > \ldots < x-n >$**)**, tj. reprezentuje seznam seznamů odpovídajících jednotlivým nezávisle proměnným.

Funkce **regression-model** má tři klíčové argumenty:

:print	s default hodnotou **t** (tisk vysledku)
:intercept	s default hodnotou **t** (regresni krivka neprochazi pocatkem souradnic)
:weights	s default hodnotou **nil** (bez pouziti statistickych vah)

Pokud chce uživatel např. zadat váhy jednotlivým pozorováním a předpokládá, že regresní křivka prochází počátkem, musí zvolit výraz

(regression-model x y :intercept nil :weights w)

kde **w** označuje proměnnou, v níž jsou uloženy příslušné váhy.

Funkce **regression-model** poskytuje uživateli souhrn základních údajů o modelu včetně odhadů regresních parametrů a jejich směrodatných odchylek, koeficientu determinace a výsledků jednoduché analýzy rozptylu.

Vytvořený regresní model je objektem (instancí prototypu **regression-proto**), a proto s ním (má-li ovšem nějaké jméno) může uživatel komunikovat pomocí posílání zpráv. Standardní implementace nabízí celkem 59 zpráv, mimo jiné:

:coef-estimates	**:coef-standard-errors**
:case-labels	**:fit-values**
:plot-residuals	**:r-squared**
:raw-residuals	**:studentized-residuals**
:residual-sum-of-squares	**:leverages**
:cooks-distances	**:anova**

Názvy těchto zpráv jsou vesměs samovysvětlující.
Poznámka.
Ve speciálním modulu **cw** od Cooka a Weisberga [3] se nachází velmi užitečná funkce **make-reg** pro lineární regresi. K vytvoření příslušného regresního objektu slouží výraz

(make-reg :data $< var - values >$ **:data-names** $< var - names >$ **:menu** $< menu - name >$ **)**

v němž $< var - values >$ je seznam seznamů tvořených hodnotami jednotlivých proměnných, $< var - names >$ seznam jmen proměnných a $< menu - name >$ název příslušného okna s menu.

Modul **cw** není součástí základní instalace, proto musí být před použitím nahrán z menu hlavního okna Lisp-Statu.

5.2 Nelineární regresní modely

Jednoduché nelineární modely lze zpracovávat pomocí funkce **nreg-model**. V tomto případě uživatel zadává výraz ve tvaru

(nreg-model $< reg - function > < y > < initial - guess >$**)**

v němž $< reg - function >$ představuje tvar teoretické regresní funkce, $< y >$ seznam hodnot závisle proměnné a $< initial - guess >$ seznam počátečních hodnot regresních parametrů.

Nechť hodnoty proměnných jsou uloženy v seznamech **x**, **y** a teoretická regresní funkce má tvar

$$\eta = \frac{\beta_0 x}{\beta_1 + x}.$$

Uživatel nejprve definuje funkci η výrazem
(def eta (beta) (/ (* (select beta 0) x) (+ (select beta 1) x)))
Jsou-li počáteční odhady parametrů $\beta_0 = 100$ a $\beta_1 = 0,1$, pak je nutno pro zpracování uvedeného modelu zadat

(nreg-model #'eta y (list 100 0.1))

Vytvořený objekt (instance prototypu **nreg-proto**) rozumí všem zprávám, které jsou srozumitelné pro objekty vytvořené pomocí funkce **regression-model**. Navíc je možno použít i dalších zpráv, např.

:count-limit	pocet iteraci (s default hodnotou 20)
:epsilon	presnost aproximace (s default hodnotou 0,0001)
:new-initial-guess	nove pocatecni odhady parametru
:parameter-names	jmena regresnich parametru

6 Ilustrativní příklad

Možnosti Lisp-Statu ukážeme na jednoduché úloze lineární regrese se dvěma nezávisle proměnnými. Vstupní data (viz tab. 1) jsou převzata z učebnice [1] a doplněna záměrně o údaje v posledním sloupci (nový bod).

Table 1: Vstupní data pro regresi

Y	4	3	4	1	6	4	5	6
x1	4	2	4	1	5	3	4	7
x2	10	8	12	3	15	12	13	10

Data uložená v souboru **"text.dat"** se načtou instrukcemi

```
(def regdata (read-data-columns "test.dat" 3))
(def x1 (select regdata 0))
(def x2 (select regdata 1))
(def y (select regdata 2))
```

K prvotnímu posouzení těchto dat poslouží funkce **scatterplot-matrix**, tedy

```
(def regscat (scatterplot-matrix (list x1 x2 y)
:title "Regression data"
:variable-labels (list "x1" "x2" "y")))
```

Podoba výsledné matice bodových grafů (viz obr. 1) ukazuje přibližně lineární závislost proměnné **y** na obou nezávisle proměnných **x1** i **x2**. (Záměrně dodaný bod je odlišen tmavým zbarvením.) Ke stejnému závěru vede i analýza grafu generovaného pomocí funkce **spin-plot**.

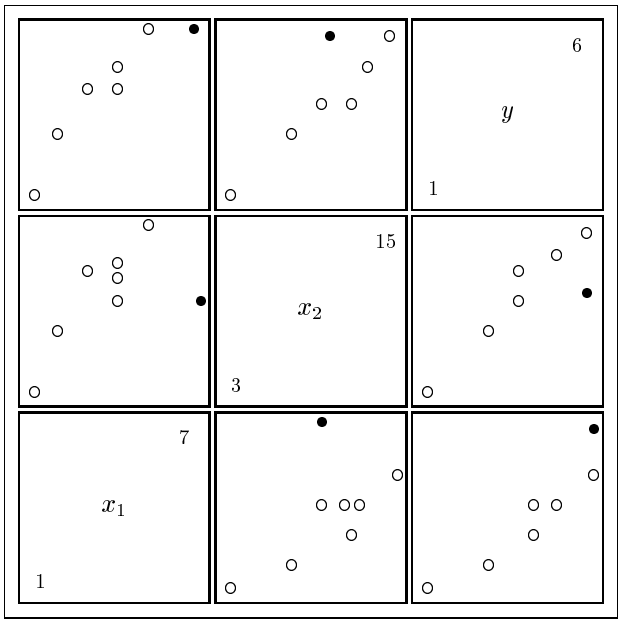

Figure 1: Matice bodových grafů pro původní data.

Lineární regresní model odpovídající teoretické regresní funkci

$$\eta = \beta_1 x_1 + \beta_2 x_2$$

vytvoříme zadáním výrazu

(def rm (regression-model (list x1 x2) y
:intercept nil))

Dostaneme následující sumární charakteristiku modelu:

```
Least Squares Estimates, Response is Y:

Label          Estimate   Std. Error    t-value
Variable 0     0.563375    0.0842835    6.68429
Variable 1     0.195238    0.0318061    6.13839

R Squared:                0.995888
Sigma hat:                0.325915
Number of cases:                 8
Degrees of freedom:              6
```

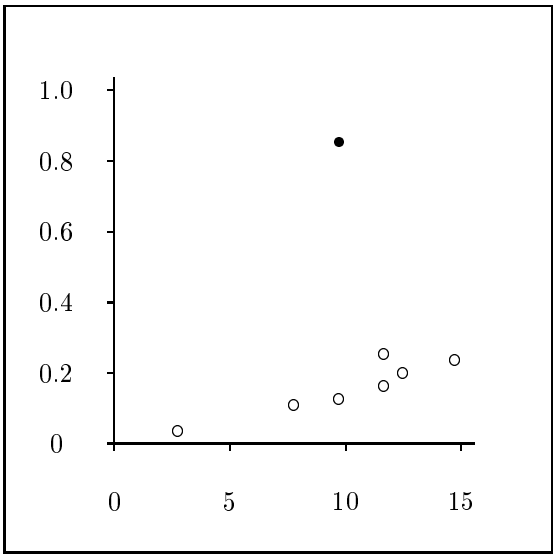

Figure 2: Diagonální prvky projekční matice pro původní data.

```
Summary Analysis of Variance Table
Source       df        SS        MS        F   p-value
Regression    2   154.363   77.1813   726.61    0.0000
Residual      6  0.637325  0.106221
```

Navržený model je na prvý pohled v pořádku, oba regresní koeficienty jsou na hladině významnosti $\alpha = 0,05$ významně odlišné od nuly.

Lisp-Stat poskytuje uživateli poměrně rozsáhlé prostředky pro regresní diagnostiku. Vedle běžných nástrojů pro analýzu reziduí (obyčejných reziduí, studentizovaných reziduí, bayesovských reziduí) jsou k dispozici i funkce **:leverages** a **:cooks-distances** pro výpočet diagonálních prvků projekční matice, resp. Cookovy vzdálenosti jednotlivých bodů. Použijme např. první z uvedených funkcí. Výrazy

```
(def lev (send rm :leverages))
(plot-points x2 lev)
```

generují grafický objekt (viz obr. 2), z něhož je zřejmé, že dodaný bod (označený tmavě) se výrazně odlišuje od všech ostatních (více než trojnásobnou hodnotou příslušného diagonálního prvku projekční matice).

Vyřaďme tedy tento bod a proveďme regresní analýzu znovu. K eliminaci uvažovaného bodu slouží např. výrazy

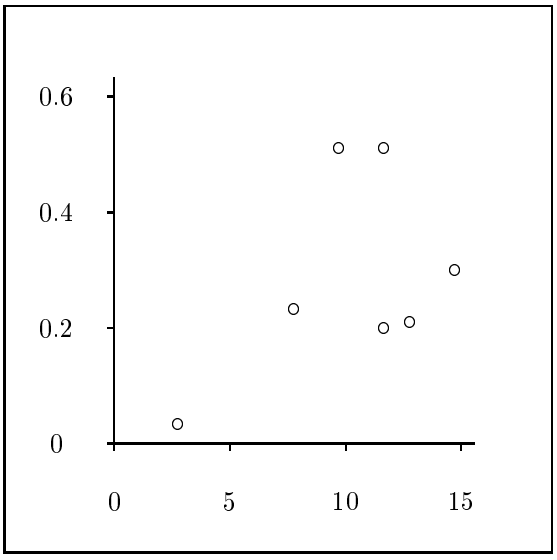

Figure 3: Diagonální prvky projekční matice po vyřazení posledního údaje.

(def x1n (select x1 (iseq 7)))
(def x2n (select x2 (iseq 7)))
(def yn (select y (iseq 7)))

Nový regresní model (v proměnných **x1n**, **x2n** a **yn**), generovaný výrazem

(def rmn ((regression-model (list x1n x2n) yn
:intercept nil))

poskytuje následující informaci:

```
Least Squares Estimates, Response is Y:
Label         Estimate   Std. Error    t-value
Variable 0    0.333333    0.247819     1.34507
Variable 1    0.266667    0.0790516    3.37332

R Squared:               0.995518
Sigma hat:               0.326599
Number of cases:                7
Degrees of freedom:             5

Summary Analysis of Variance Table
```

```
Source       df        SS        MS        F   p-value
Regression    2   118.467   59.2333   555.31    0.0000
Residual      5  0.533333  0.106667
```

Po vyřazení záměrně přidaného bodu se situace změnila. Proměnná **y** (nyní označena **yn**) závisí i nadále významně na vysvětlující proměnné **x2** (**x2n**), ale její závislost na **x1** (**x1n**) není prokazatelná ($1,34507 < t_5(0,05)$). Diagonální prvky nové projekční matice jsou zobrazeny na obr. 3.

7 Závěr

V příspěvku jsou stručně popsány základní instrukce verze XLISP-STAT prostředí Lisp-Stat a posouzeny její možnosti při analýze dat. Jde o prostředí moderní (objektově orientované), flexibilní (otevřené a rozšiřovatelné) a snadno dostupné pro každého uživatele. Dosavadní zkušenosti nasvědčují tomu, že je mimořádně vhodné pro průzkumovou analýzu dat (viz např. [2]). Snad jedinou nevýhodou je poněkud nezvyklý zápis instrukcí (prefixová notace), na nějž si lze ovšem brzy zvyknout.

Základní modul XLISP-STATu, jehož tvůrcem je Luke Tierney [5], je volně k dispozici na adrese

http://umnstat.stat.umn.edu

V současnosti již existuje celá řada doplňujících modulů, jež jsou také bezplatně distribuovatelné. Nejznámější je patrně knihovna modulů, kterou udržuje Jan de Leeuw [4]. Tato knihovna obsahuje mimo jiné již uvedený modul **cw** pro regresní analýzu a dále moduly pro analýzu kategoriálních dat, mnohorozměrnou statistickou analýzu, metody Monte Carlo, robustní statistiku a analýzu časových řad. Uživatel může jednotlivé moduly získat na adrese

http://www.stat.ucla.edu

Na tomto místě je vhodné zmínit se o vizuálním statistickém systému **ViSta**, navrženém F. W. Youngem [6] nejen pro profesionální statistickou práci, ale také pro výuku statistiky. Základní informace o tomto systému včetně dokumentace a programových modulů jsou volně přístupné na adrese

http://forrest.psych.unc.edu

References

[1] J. Anděl: *Matematická statistika*. Praha, SNTL 1978.

[2] A. Bartkowiak: *Exploratory Data Analysis, its Historical Development, what it is Today*. Biocybernetics and Biomedical Engineering, **15**, 1-2, 1995, 85−120.

[3] R. D. Cook, S. Weisberg: *An Introduction to Regression Graphics*. New York, Wiley 1994.

[4] J. de Leeuw: *The Lisp-Stat Statistical Environment*. Statistical Computing & Graphics, **5**, 3, 1994, 13−17.

[5] L. Tierney: *LISP-STAT . An Object-Oriented Environment for Statistical Computing and Dynamic Graphics*. New York, Wiley 1990.

[6] F. W. Young, R. A. Faldowski, M. M. McFarlane: *Multivariate Statistical Visualization*. In: C. R. Rao (Editor) Handbook of Statistics, **9**, 1993, 959−998.